79

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

Facultad de Ingeniería

PAQUETE DE CONTROLADORES DIGITALES MEDIANTE PROGRAMACION ORIENTADA A OBJETOS. CDOBJECT

T E S I S

Que para obtener el Título de:

INGENIERO EN COMPUTACION

p r e s e n t a

ARMANDO ROSAS MORATO

DIRECTOR DE TESIS: ING. RICARDO GARIBAY JIMENEZ

México, D. F. 1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

A mis padres :

Sra. Ramona Morato de Rosas
Ing. Eusebio Rosas Chapa

con gran cariño y admiración,
por haber enriquecido
abnegadamente mi vida
con amor, apoyo y ejemplo sin par.

A mis compañeros y amigos :
por la alegría contagiosa con que llenaron mis años de estudio.

A mis maestros :
por la invaluable contribución dada a mi formación.

A la Facultad de Ingeniería U.N.A.M.:
con eterno agradecimiento .

I N D I C E :

CAPITULO I

INTRODUCCION

Como factor fundamental e integral de los modernos procesos industriales y de manufactura, el control automático posee una intervención cada vez más importante en la vida diaria. Cumple un papel importante en la operación de plantas dedicadas a la producción de bienes de consumo en general.

A medida que se ha perfeccionado, la computadora digital ha llegado a adquirir importancia trascendental dentro de las áreas de la actividad humana. Por lo tanto no podía dejar de ser un instrumento fundamental para la realización de las tareas de control. Debido a su versatilidad, la computadora digital hace posible introducir técnicas de control que eran posibles con instrumentación analógica, y también aquellas cuya aplicación con dicho tipo de instrumentación no podían realizarse.

En prácticas y proyectos del Laboratorio de Control Digital de la Facultad de Ingeniería se han desarrollado y empleado paquetes de programación que permiten la aplicación de algoritmos de control en lazo cerrado a procesos continuos. Las primeras versiones de este software permitían aplicar acciones PID y elementales funciones de interface con el operador y con el proceso a controlar.

La versión más actual de dichos paquetes de control, denominada como "PCDigital", ha mejorado la interface con el usuario; sin embargo, adolece de ciertas deficiencias. Ellas son :

* Carece de un diseño estructurado que permita el mantenimiento así como su crecimiento. A pesar de estar codificado en Lenguaje Pascal, no se hizo uso de la potencia para estructuración que permite tal lenguaje.
* Posee código redundante como consecuencia de la característica anterior. Por lo tanto aumenta el tamaño del programa.

* Carece de una Guía de Mantenimiento.
* No posee manejo de errores por parte del programa, lo cual puede provocar problemas al tiempo de corrida.
* Emplea dos tarjetas de adquisición de datos, PCL-812 y PCL-726, haciendo cada una de ellas por separado las conversiones A/D y D/A respectivamente.

Por las razones ya expuestas, se desarrolló un programa en lenguaje C++, empleando la metodología de programación orientada a objetos (POO), con el fin de superar las deficiencias de programación que "PCDigital" tenía .

La metodología de POO es una nueva forma de enfocar el trabajo de programación. Tomando las mejores ideas de la programación estructurada, le permite al programador descomponer un problema en subgrupos de partes relacionadas. Estos subgrupos se traducen en unidades contenidas en sí mismas llamadas objetos.

Todos los lenguajes de programación orientada a objetos poseen tres cosas en común : Objetos, Polimorfismo y Herencia. Un Objeto es una entidad lógica que contiene tanto los datos como el código que manipula los mismos, pudiendo ser inaccesibles o no tanto uno como otro a algo fuera del objeto. De ésta forma, un objeto provee un significativo nivel de protección en contra de modificaciones accidentales o uso incorrecto. En suma, un Objeto es un tipo de variable definido por el usuario.

El polimorfismo le permite a un nombre ser usado para propósitos relacionados pero ligeramente diferentes. Su propósito es especificar una clase general de acción con un sólo nombre. Dependiendo del tipo de dato con el que se trata, un caso específico del caso general es ejecutado.

La herencia o sucesión es el proceso mediante el cual un objeto puede adquirir las propiedades de otro objeto, apoyando el

concepto de clasificación. Esto provoca que la información sea manejable mediante clasificaciones jerárquicas. Sin el uso de clasificaciones, cada objeto tendría que definir todas sus características explícitamente. Con las clasificaciones, un objeto sólo necesita definir aquellas cualidades que lo hacen único dentro de su clase.

La metodología POO requiere que, en primera instancia se establezcan los datos así como las acciones (funciones) que operan sobre los mismos,y que puedan agruparse dentro de un mismo objeto merced a sus características. Es decir, aquellos datos y código que poseen afinidad entre sí pueden formar una entidad lógica que en C++ se define como "Clase". Asimismo se establecen las reglas de herencias entre "Clases", las cuales permiten que elementos de una(s) posean propiedades de otra(s).

Dependiendo de los requerimientos del software a desarrollar pueden hacerse uso o no, según sea el caso, de otras características de C++, las cuales se citan a continuación:

* Constructores y Destructores. Son funciones que se ejecutan al momento de la creación y destrucción del objeto respectivamente. Dado que el objeto se crea de una clase al momento de ejecución, los Constructores y Destructores son usados normalmente como subrutinas de inicialización y terminación.
* Sobrecarga (Overloading) de funciones y operadores. Permite el polimorfismo en funciones, incluyendo Constructores, así como en los operadores. Los operadores son un tipo de funciones que permiten hacer operaciones entre los datos de objetos distintos pero de misma clase.
* Inicialización Dinámica. Es posible inicializar variables locales y globales al tiempo de corrida.
* Funciones Virtuales. Funciones que son declaradas en una clase base y son redefinidas en clases derivadas. Al ser accesadas mediante un apuntador a la clase base, C++

determina cual función es llamada. Implica polimorfismo al tiempo de ejecución.

Sin llegar a emplear algunos de los factores ya citados,debido a que no existia la necesidad de ellos,se obtuvo el software de nombre "CDobject", el cual además de superar las deficiencias de "PCDigital" presenta las siguientes mejoras:

* Opciones del sistema, como visualización de directorios, cambio de directorio, copiado de archivos, etc.
* Ayuda al momento de la corrida.
* Interfaz sistema-usuario muy amigable. Soporta Mouse y hace uso extensivo de ventanas y gráficos elaborados.
* Asignación dinámica de la memoria para Objetos, estructuras de datos y en general para cualquier tipo de dato cuya presencia a todo lo largo de la ejecución no sea necesaria.
* Historia de control de proceso.
* Aplicación ininterrumpida de acción de control al cambiar parámetros del controlador o solicitar historia de control de proceso.
* Generación avanzada de código ejecutable para microprocesador 80286, con opción a usar coprocesador matemático 80x87.
* Estructura de Proyecto, lo que facilita la localización de errores y reduce el tiempo de compilación.

Conceptos relativos a Control Digital, estructura general del programa, pruebas del mismo, etc, son mostrados en el presente trabajo.

CAPITULO II

ALGORITMOS DE CONTROL DIGITAL

a) Conceptos Generales :

El planteamiento básico de un sistema de control digital se expone en la figura II.1.

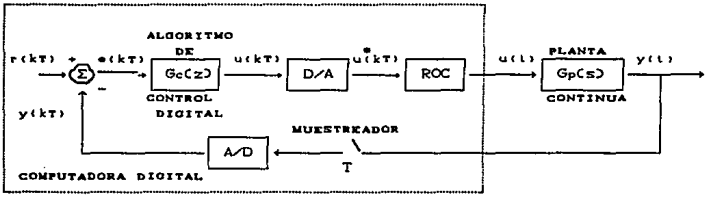


FIGURA II.1 Esquema de Control Digital.

Estableciendo las siguientes definiciones :

Gp(s) Función de transferencia continua de la planta o proceso.

Gc(z) Función de transferencia discreta del controlador

r(kT) Señal discreta de referencia.

e(kT) Secuencia discreta de error.

u(kT) Secuencia discreta de control.

y(t) Salida de la planta.

Como puede observarse, se forma un lazo de control simple en el que la computadora forma parte de la trayectoria directa del lazo, actuando sobre la planta y con realimentación unitaria. La computadora compara el valor efectivo de salida de la planta con el valor deseado (señal de referencia), produciendo una señal de control que reduce la desviación (error) a cero o a un valor pequeño.

Los controladores digitales poseen excelentes ventajas sobre los de tipo analógico. Una de ellas es que pueden realizar complejos cálculos con gran exactitud a alta velocidad con un costo relativamente pequeño. Otra es que son extremadamente versátiles. Simplemente colocando un nuevo programa se pueden cambiar las operaciones a efectuar.

En el análisis y diseño de un controlador digital, generalmente se lleva a cabo el siguiente procedimiento :

I) Obtención de un sistema equivalente discreto $G_p(z)$ de la planta continua $G_p(s)$. Este equivalente discreto se calcula mediante la técnica de Retén de Orden Cero, cuya expresión es:

$$G_p(z) = (1 - z^{-1}) \mathcal{Z} \{G_p(s)/s\}$$

II) Dada una estructura y parámetros de un controlador analógico $G_c(s,\theta')$, donde θ' representa los parámetros de diseño, se obtiene mediante aproximación un controlador discreto $G_c(z,\theta)$ donde $\theta = f(\theta',T)$. Esta aproximación discreta del controlador analógico puede realizarse mediante distintos métodos, los cuales, en general, son reemplazos de s por una expresión de z.

III) Ya obtenida la estructura del controlador discreto $G_c(z,\theta)$, se procede a expresarla como una ecuación en diferencias, a fin de editarla como un programa de computadora, en el lenguaje deseado para posteriormente poder ser ejecutado y aplicado al proceso en cuestión.

El segundo paso no siempre es necesario ya que disponiendo de la descripción discreta de la planta, es posible determinar directamente la estructura del controlador discreto, mediante la especificación de un objetivo de

control adecuado.

b) Equivalentes discretos de Acciones Controladoras :

La teoría de control considera tres acciones básicas de control : la proporcional, la integral y la derivativa. Estas acciones se pueden aplicar de manera combinada dando lugar a tres de los más populares algoritmos de control : el controlador proporcional (P), el proporcional-integral (PI) y el proporcional-integral-derivativo (PID). Su importancia radica en que no solo son tradicionalmente estudiados en los primeros cursos de control automático sino que también dan solución satisfactoria a múltiples aplicaciones en la industria.

Por otro lado, la aparición de sistemas de control digital han impulsado el desarrollo de nuevas técnicas de control cuya aplicación es exclusiva de ellos, como lo son los obtenidos por métodos de Asignación de Polos así como los del tipo autosintonizable.

b.1) Controlador Proporcional P :

Al controlador proporcional le corresponde ser uno de los más simples algoritmos de control. La obtención de su ecuación en diferencias, así como su implantación en la computadora así lo demuestran.

La función de transferencia de un controlador proporcional analógico está dada por la expresión (1) .

$$G_c(s) = \frac{U(s)}{E(s)} = K \qquad (1)$$

donde :

s	Variable de Laplace
U(s)	Transformada de Laplace de la señal de control
E(s)	Transformada de Laplace de la señal de error

K Ganancia del controlador

Al no existir términos en 's' la discretización es :

$$Gc(z) = \frac{U(z)}{E(z)} = K \qquad (2)$$

Despejando U(z) y antitransformando se obtiene la ecuación en diferencias de la acción P :

$$u(kT) = K\, e(kT) \qquad (3)$$

Dado que este algoritmo actúa como un amplificador de la señal e(t), puede mejorar la respuesta transitoria, no así la permanente pues ésta última puede poseer un sesgo no nulo en el error. Para resolver este problema es necesaria la aplicación de algoritmos más elaborados.

b.2) Controlador Proporcional-Integrativo PI :

Ligeramente más complejo que el P, pero con mayores ventajas, es el algoritmo de control proporcional-integrativo. Su introducción en un lazo de control proporciona la eliminación del error en estado estacionario para cambios en la referencia y eventuales perturbaciones aditivas constantes. Empero, debe considerarse que la introducción del término integrativo produce disminución en los márgenes de estabilidad cuando se trata de plantas de orden elevado.

La función de transferencia dada por la expresión (4) define al controlador PI analógico en el dominio de la variable de Laplace.

$$Gc(s) = \frac{U(s)}{E(s)} = K\left(1 + \frac{1}{Ti\,s}\right) \qquad (4)$$

Además de los términos ya enunciados, apreciamos en (4) el parámetro Ti, el cual se define como :

Ti Constante de tiempo de integración.

La aproximación discreta de PI se hace mediante el método de "aproximación rectangular en adelanto", el cual propone que la variable s se aproxime a la variable z mediante la siguiente expresión :

$$s \simeq \frac{z - 1}{T} \qquad (5)$$

Sustituyendo (5) en (4) obtenemos la función de transferencia del controlador PI discreto en el dominio de la variable z :

$$Gc(z) = \frac{KTi(1 - z^{-1}) + KTz^{-1}}{Ti(1 - z^{-1})} \qquad (6)$$

Por último, la ecuación en diferencias que define el algoritmo de control se obtiene al despejar y antitransformar la expresión (6).

$$u(kT) = u(kT-T) + Ke(kT) + K(\frac{T}{Ti} - 1)e(kT-T) \qquad (7)$$

b.3) Controlador Proporcional-Integral-Derivativo PID

El controlador de empleo más frecuente en diversas aplicaciones es el proporcional-integral-derivativo. Debido a que conjunta las tres acciones de control mencionadas, presenta ventajas sobre los algoritmos anteriores mejorando la precisión y la estabilidad del sistema en malla cerrada.

La ecuación (8) enunciada a continuación describe al controlador PID tradicional en su forma continua .

$$G_c(s) = \frac{U(s)}{E(s)} = K\left(1 + \frac{1}{T_i s} + \frac{T_d s}{1 + (T_d/N)s}\right) \qquad (8)$$

donde, además de los términos enunciados en las ecuaciones (1) y (4) se tiene un nuevo parámetro : la constante de tiempo derivativo Td.

En la expresión anterior se aprecia que la acción derivativa, correspondiente al término Tds, posee además un filtro paso-bajas con una constante de tiempo Td/N. Sin el filtro, la acción derivativa sería no causal y por lo tanto físicamente irrealizable.

La aproximación discreta de la acción PID se efectúa empleando el método de "aproximación rectangular en adelanto" (5) para la acción integrativa, y la "aproximación rectangular en atraso" (9) para la acción derivativa y el filtro.

$$s \simeq \frac{z - 1}{Tz} \qquad (9)$$

Las equivalencias resultantes son :

	Forma continua	Forma discreta
acción P	K	K
acción I	$\dfrac{K}{T_i s}$	$\dfrac{\alpha z^{-1}}{(1 - z^{-1})}$
acción D	$KT_d s$	$\dfrac{1}{\beta}(1 - z^{-1})$
filtro	$\dfrac{1}{1 + (T_d/N)}$	$\dfrac{1}{\gamma - (\gamma - 1)z^{-1}}$

Los parámetros "α", "β" y "γ" están dados por las expresiones :

$$\alpha = \frac{KT}{T_i} \qquad \beta = \frac{T}{KT_d} \qquad \gamma = \frac{T_d}{NT} + 1 \qquad (10)$$

La ecuación en diferencias total que define al algoritmo de control se obtiene adicionando las componentes enunciadas de acuerdo con las estructuras mostradas mediante diagramas de bloques de la figura II.2.

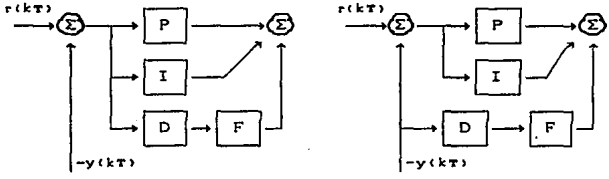


Figura II.2 Estructuras de acción PID .

Cada una de las estructuras mostradas dan origen a distintas ecuaciones en diferencias, las cuales pueden representarse por una ecuación general de la forma :

$$U(z) = \frac{T(z)}{S(z)} R(z) - \frac{Q(z)}{S(z)} Y(z) \qquad (11)$$

siendo :

$$S(z) = s_0 + s_1 z^{-1} + s_2 z^{-2} = \gamma + (1 - 2\gamma) z^{-1} + (\gamma - 1) z^{-2} \qquad (12)$$

$$Q(z) = q_0 + q_1 z^{-1} + q_2 z^{-2} \qquad (13)$$

$$q_0 = K\gamma + 1/\beta$$
$$q_1 = K(1 - 2\gamma) + \alpha\gamma - 2/\beta$$
$$q_2 = (K - \alpha)(\gamma - 1) + 1/\beta$$

En el caso del presente trabajo sólo tomaremos en cuenta el planteamiento de la estructura (b). Para ésto, el polinomio T(z) se define como:

$$T(z) = t_0 + t_1 z^{-1} + t_2 z^{-2} \qquad (14)$$

$$t_0 = K\gamma$$
$$t_1 = K(1 - 2\gamma) + \alpha\gamma$$
$$t_2 = (K - \alpha)(\gamma - 1)$$

Sustituyendo S(z), T(z) y Q(z), despejando y antitransformando se obtiene la ecuación en el dominio del tiempo discreto para la estructura PID-(b).

$$s_0 u(kT) = - s_2 u(kT-2T) - s_1 u(kT-T) + t_0 r(kT) + t_1 r(kT-T) + t_2 r(kT-2T) - q_2 y(kT-2T) - q_1 y(kT-T) - q_0 y(kT)$$

Ecuacion (15)

b.4) Controlador por el Método de Asignación de Polos

Anteriormente se señaló que en el proceso de análisis y diseño de un controlador digital no siempre es necesario emplear métodos de aproximación discreta. Disponiendo de la descripción discreta de la planta se puede determinar la estructura del controlador mediante la especificación del objetivo de control. De acuerdo con este tipo de procedimiento es el método de asignación de polos.

Consiste en un conjunto de reglas las cuales, a partir de la función de transferencia Gp(z) del proceso y especificaciones en términos del comportamiento dinámico

deseado en malla cerrada, permiten obtener una estructura lineal de control. A continuación se presenta su planteamiento general así como la solución de un caso específico.

El modelo discreto de la planta o proceso está dado por :

$$Gp(z) = \frac{B(z)}{A(z)} = \frac{Y(z)}{U(z)} \qquad (16)$$

donde A(z) y B(z) son polinomios coprimos, es decir, sin factores comunes. Y(z) y U(z) son las transformadas $\mathcal{Z}$ de la variable del proceso y señal de control respectivamente. La correspondiente función de malla cerrada se enuncia como :

$$Hm(z) = \frac{Bm(z)}{Am(z)} = \frac{Y(z)}{R(z)} \qquad (17)$$

donde R(z) es la transformada $\mathcal{Z}$ de la entrada de referencia.

El método propone la siguiente ecuación como ley de control:

$$U(z) = \frac{T(z)}{S(z)} R(z) - \frac{Q(z)}{S(z)} Y(z) \qquad (18)$$

donde S, T y Q son polinomios. La figura II.3 muestra la estructura del sistema.

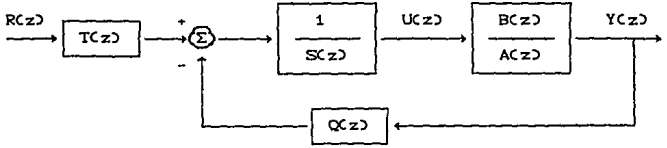


Figura II.3 Estructura de acción por Método de Asignación de Polos

Se asume que S es mónico, es decir, que el coeficiente

de la potencia mayor en S es unitario. La ecuación (18) representa una combinación de la señal de referencia con la función de transferencia :

$$H_{rf}(z) = \frac{T(z)}{S(z)} \qquad (19)$$

y de la salida "Y" con la función de transferencia :

$$H_{rb}(z) = \frac{Q(z)}{S(z)} \qquad (20)$$

La ley de control dada por (18) requiere de condiciones que aseguren que (19) y (20) sean causales, siendo aquellas:

$$\text{grad } S \Rightarrow \text{grad } T \qquad (21)$$
$$\text{grad } S \Rightarrow \text{grad } Q \qquad (22)$$

Además, si se considera que el tiempo de cálculo de la señal de control es despreciable con respecto al período de muestreo, entonces es natural requerir que :

$$\text{grad } S = \text{grad } T = \text{grad } Q \qquad (23)$$

Por otro lado, si el tiempo de computación es cercano al del período de muestreo, entonces la correspondiente restricción es :

$$\text{grad } S = 1 + \text{grad } T = 1 + \text{grad } Q \qquad (24)$$

Despejando U(z) de (18) y sustituyéndola en (16), tenemos que :

$$H_m(z) = \frac{B_m(z)}{A_m(z)} = \frac{B(z)T(z)}{A(z)S(z) + B(z)Q(z)} \qquad (25)$$

El problema de diseño se reduce a encontrar los

polinomios S, Q y T que satisfagan (25) y requerimientos adicionales.

Para ésto se plantea que :

* Si un factor de B no es un factor de B_m, entonces debe serlo de AS+BQ, por lo tanto debe ser cancelado por un polo de lazo cerrado.
* Como el sistema de lazo cerrado debe ser estable, se deduce que sólo ceros estables pueden ser cancelados. Así, factorizamos a B como :

$$B = B^+B^- \qquad (26)$$

donde B^- tiene todos sus ceros fuera del disco unitario, mientras que B^+ los tiene dentro.

* Para obtener una factorización única de B, el coeficiente del término de mayor exponente en B^+ es fijado a uno. Se dice entonces que el polinomio B^+ es mónico.
* Ya que B^- no es factor de AS+BQ, entonces debe serlo de B_m :

$$B_m = B^-B_m' \qquad (27)$$

Esto implica que los ceros inestables del proceso no pueden ser cambiados, pero sí deben ser incluídos en B_m.

* Como B^+ es factor de AS+BQ, se deduce que también lo es de S :

$$S = B^+S' \qquad (28)$$

* Sustituyendo (26), (27) y (28) en (25), se tiene que :

$$\frac{B^+B^-T}{B^+(AS' + B^-Q)} = \frac{B^-B_m'}{A_m} \qquad (29)$$

* Simplificando (29) se obtiene la siguiente expresión :

$$\frac{T}{AS' + B^-Q} = \frac{B_m'}{A_m} \qquad (30)$$

* Factorizando T como :

$$T = Bm'Ao \qquad (31)$$

donde Ao se conoce como polinomio observador. Despejando ambos lados de la ecuación (30) y sustituyendo (31) en aquella, la expresión obtenida es:

$$AS' + B^{-}Q = AoAm \qquad (32)$$

* Tomando en cuenta (32) se aprecia que :

$$\text{grad } S' + \text{grad } A = \text{grad } Ao + \text{grad } Am \qquad (33)$$

$$\text{grad } S' = \text{grad } Ao + \text{grad } Am - \text{grad } A \qquad (34)$$

* Por último, las expresiones siguientes señalan otras condiciones de compatibilidad .

$$\text{grad } A - \text{grad } B <= \text{grad } Am - \text{grad } Bm \qquad (35)$$

$$\text{grad } Ao = 2\text{grad } A - \text{grad } Am - \text{grad } B^{+} - 1 \qquad (36)$$

En síntesis, el procedimiento de análisis y diseño es :

A) Conocer el modelo del proceso mediante $G_p(z)$, así como especificaciones dadas por Am y Bm. Verificar (35).
B) Factorizar B en (26) tomando en cuenta (27), y obtener el grado de Ao según ecuación (36).
C) Definir el grado de Q y S' mediante :

$$\text{grad } Q = \text{grad } A - 1 \qquad (37)$$

y (34) respectivamente.
D) Resolver la ecuación (32) cuya solución será S' y Q,

habiendo propuesto Ao siendo éste mónico y estable.

E) Ya conocidos S' y Q se calcula S mediante (28) así como T mediante (31).

F) Se define U(z) y se procede a obtener la correspondiente ecuación en diferencias u(k).

Como ejemplo de aplicación, se establece el siguiente problema. La función de transferencia de un proceso de 2° orden está dada por la ecuación (38).

$$Gp(z) = \frac{K(z - b)}{(z - a)(z - c)} \qquad (38)$$

Se asume que la función de malla está caracterizada por:

$$Hm(z) = \frac{z(1 + p_1 + p_2)}{z^2 + p_1 z + p_2} \qquad (39)$$

estando p_1 y p_2 definidos por :

$$p_1 = -2e^{-\xi\omega nT}\cos\left(\omega n\sqrt{1 - \xi^2}\,T\right) \qquad (40)$$

$$P_2 = e^{-2\xi\omega nT} \qquad (41)$$

donde : ξ es el coeficiente de amortiguamiento.
ωn es la frecuencia natural del proceso.

La función de transferencia Gp(z) tiene un cero en z = b el cual no se encuentra incluído en Hm(z), por lo cual es necesario cancelarlo. Factorizando B como :

$$B = B^+B^-$$
$$B^+ = (z - b) \qquad (42)$$
$$B^- = K \qquad (43)$$

Entonces por condición de compatibilidad dada por (27),

se propone :

$$Bm' = \frac{Bm}{K} = \frac{z(1 + p_1 + p_2)}{K} \qquad (44)$$

Se emplea (36) para obtener el grado del polinomio observador.

$$\text{grad } Ao = 2\text{grad } A - \text{grad } Am - \text{grad } B^+ - 1$$
$$\text{grad } Ao = 4 - 2 - 1 - 1 = 0$$
$$\text{grad } Ao = 0$$

Como el grado de Ao es cero, entonces se escoge que :

$$Ao(z) = 1 \qquad (45)$$

Con las ecuaciones (34) y (37) se obtienen los grados de los polinomios S' y Q respectivamente.

$$\text{grad } S' = \text{grad } Ao + \text{grad } Am - \text{grad } A$$
$$\text{grad } S' = 0 + 2 - 2 = 0$$

$$\text{grad } Q = \text{grad } A - 1 = 2 - 1 = 1$$

En base a los resultados anteriores se tiene que :

$$S'(z) = so \qquad (46)$$
$$Q(z) = qoz + q_1 \qquad (47)$$

Sustituyendo (43),(45), (46) y (47) en (32) .

$$(z - a)(z - c)so + K(qoz + q_1) = z^2 + p_1z + p_2 \qquad (48)$$

Con (48) se forma el siguiente sistema de ecuaciones lineales :

$$so = 1 \qquad (49)$$

$$-(a + c)s_o + Kq_o = p_1 \qquad (50)$$

$$acs_o + Kq_1 = p_2 \qquad (51)$$

Resolviendo el sistema formado por (49), (50) y (51) :

$$s_o = 1 \qquad (52)$$

$$q_o = \frac{p_1 + a + c}{K} \qquad (53)$$

$$q_1 = \frac{p_2 - ac}{K} \qquad (54)$$

Con los resultados anteriores se procede a obtener S y T mediante (28) y (31) respectivamente :

$$S(z) = B^{+}S' = (z - b) \qquad (55)$$

$$T(z) = B_m'A_o = \frac{z(1 + p_1 + p_2)}{K} \qquad (56)$$

Sustituyendo (47), (55) y (56) en la ecuación (18) :

$$(z - b)\,U(z) = \frac{z(1+p_1+p_2)}{K} R(z) - \left[\frac{(p_1+a+c)z}{K} + \frac{(p_2-ac)}{K}\right] Y(z)$$

Ecuacion (57)

Antitransformado (57) y despejando u(k) se obtiene :

$$u(k) = \left[\frac{1 + p_1 + p_2}{K}\right] r(k) - \left[\frac{p_1 + a + c}{K}\right] y(k) -$$

$$- \left[\frac{p_2 - ac}{K}\right] y(k-1) + bu(k-1) \qquad (58)$$

que es la ley de control obtenida con el método de asignación de polos con cancelación de ceros. La siguiente figura muestra la solución al ejemplo anterior mediante diagramas de bloques.

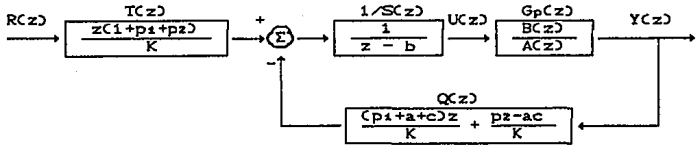


Figura II.4 Solución al ejemplo del proceso de 2° orden.

CAPITULO III

ARQUITECTURA DEL PROGRAMA

a) Especificaciones .

a.1) Enunciado del problema :

Desarrollar un programa en lenguaje C++, empleando la metodología de programación orientada a objetos, que permita controlar en lazo cerrado diversos procesos continuos. El paquete debe ser estructurado modularmente a fin de que permita su crecimiento posterior para la aplicación de algoritmos de control avanzado.

a.2) Funciones proporcionadas :

La ejecución del paquete debe alternarse entre la operación en línea y fuera de línea. En línea la computadora actúa como controlador digital, efectuando el procesamiento de la información en tiempo real. A continuación se presentan las funciones proporcionadas bajo cada tipo de operación así como aquellas que están presentes durante toda la ejecución.

* Durante la operación en línea :

(i) Empleo de la Tarjeta de adquisición de dato PCL-812 tanto para conversiones A/D como D/A.

(ii) Las acciones de control :

Manual .
Proporcional .
Proporcional-Integrativo .
Proporcional-Integral-Derivativo .
Asignación de Polos con cancelación de Ceros.

(iii) Historia de Control de Proceso.

(iv) La aplicación de la accion de control no debe interrumpirse al cambiar parámetros del controlador o al solicitar historia de control de proceso.

* Durante la operación fuera de línea :

Opciones de apoyo al usuario tales como :

- Ayuda para manejar el paquete.

- Operaciones de disco como visualización de directorios, cambio de éstos y copiado de archivos

* Bajo ambos tipos de operación :

(i) Interfaz sistema-usuario amigable. Implica el uso extensivo de gráficos de alta resolución para la presentación de menús, submenús, ventanas para entrada/salida de datos así como pantallas de graficación de señales del proceso. También debe soportar Mouse .

(ii) Detección y corrección de errores .

a.3) Ambiente de procesamiento Hardware y Software :

Las características básicas de Hardware son :

* Computadora 316 AT con μprocesador 80386 y coprocesador 80387, memoria RAM de 2 Mbytes y señal de reloj a frecuencia de 16 Mhz.
* Monitor VGA a colores .
* Tarjeta de adquisición de datos PCL-812.
* Mouse con software de control propiamente instalado.
* Disco duro de 80 Mbytes.

La ejecución del paquete se dará bajo ambiente del sistema operativo DOS versión 5.0 .

b) Diagramas de estructura .

Previo a la programación, es necesario conocer con detalle la naturaleza del software a desarrollar. Para ésto hay que diferenciar y caracterizar los elementos o módulos principales que componen el programa .

En base a especificaciones mencionadas y características de un sistema de control digital, es posible dividir el programa en cinco módulos o tareas :

(i) De comunicaciones . Encargado de la Entrada/Salida de datos del controlador . Esto es, inicializar la operación de la tarjeta PCL-812, seleccionar canales de entrada y salida analógicos, efectuar conversiones A/D y D/A y por último deshabilitar su operación.

(ii) De algoritmos de control. Conjunta la inicialización y validación de parámetros de controladores, las ecuaciones en diferencias que definen a éstos y la traslación en el tiempo de valores de señales del controlador.

(iii) De interface sistema-usuario. Hace uso de dispositivos de Entrada/Salida del usuario como son el monitor, teclado y mouse. En primera instancia establece subrutinas de inicialización, aplicación y terminación para cada dispositivo. Las de aplicación comprenden el uso de primitivos de gráficos para el monitor; de funciones dedicadas a la lectura y validación de datos numéricos y alfanuméricos del teclado; de funciones encargadas de informar la posición del mouse así como el sensado de sus botones. Después, con las anteriores subrutinas empleadas en forma coordinada, se ofrece al usuario una interface integral.

(iv) De apoyo al sistema. Comprende subrutinas que verifican el estado de las unidades del disco, leen directorios, hacen cambios de ellos y copian archivos.

(v) De apoyo al usuario. Consta de funciones que leen un archivo en disco y permiten desplegarlo en pantalla durante la operación fuera de línea. El archivo contiene información para el manejo del paquete.

Para su operación cada módulo requiere una serie de datos. Estos son proporcionados por otro(s) módulo(s). Las relaciones que facilitan la transferencia de información entre módulos se muestran en la figura III.1 .

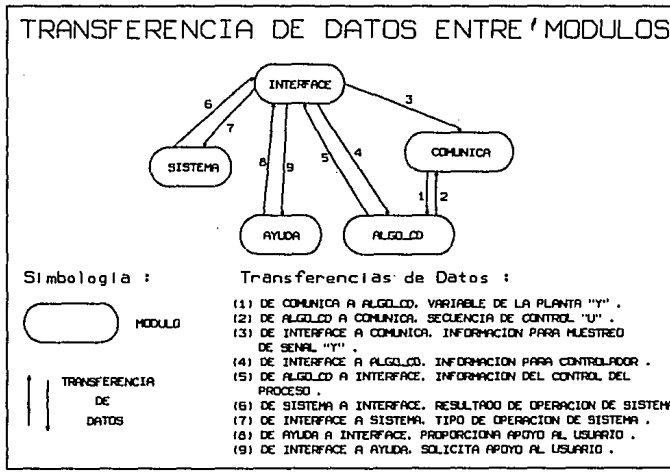


FIGURA III.1 Transferencia de Datos entre módulos .

La figura anterior muestra con claridad la fuerte interdependencia dada entre los módulos para su operación en conjunto como programa.

Retomando los conceptos de la programación orientada a objetos, se plantea la estructura modular como una estructura formada por clases. Dado que una clase es una entidad lógica de código-dato inherente a un elemento diferenciable y

caracterizable, se clasifican los elementos susceptibles de formarlas . Por otro lado , se definen también las partes de la clase que son privadas a ella así como aquellas que son accesibles a otras clases, mediante herencia, ó al programa principal.

Dentro del módulo de comunicaciones se estableció que éstas se darían, en primer momento, mediante la tarjeta de adquisición de datos PCL-812, y a futuro, a través del puerto RS-232. Entonces se propone una estructura de clases formada por dos raíces y un nodo ó unión. Las clases raíz son : Tarjeta y Puerto, correspondiendo respectivamente a la tarjeta PCL-812 y al puerto RS-232. La clase nodo se denomina Comunica y es la encargada de proporcionar acciones genéricas de Tarjeta y Puerto .

Para el módulo de algoritmos de control se definen tantas clases como controladores se tengan además de una clase nodo. Las clases que corresponden a los algoritmos de control manual, p, pi, pid, por asignación de polos y autosintonizable,son : Manual, P, Pi, Pid, Apolos y Autosin . Estas clases reunen sólo el código del controlador, a excepción de Autosin que sólo es enunciada. La clase nodo , llamada Algo_cd, contiene señales y parámetros de los controladores, conjuntándolos con acciones genéricas de Comunica a través de la herencia de ésta última.

El módulo de Interface sistema-usuario comprende las clases raíz Monitor, Mouse, Teclado y como clases nodo a Interface1 e Interface2. Cada una de las clases raíz posee datos y código para el funcionamiento y control de cada dispositivo de E/S mencionado en especificaciones. Interface1 hereda las clases raíz de su módulo junto con la clase Algo_cd. De ésta forma, Interface1 proporciona una interface sistema-usuario integral para la transferencia de

información entre el usuario, módulos de comunicaciones y de algoritmos de control.

Los módulos de apoyo al sistema y de apoyo al usuario forman por separado las clases Sistema y Ayuda. Estas son heredadas junto con Interface1 por la clase Interface2, cuya tarea es similar a la realizada por Interface1 .

La clase Interface2 hereda a la clase Cdobject los elementos públicos de todas las clases. Como clase nodo principal, Cdobject posee subrutinas que realizan las operaciones en y fuera de línea.

En suma :

* Cada clase representa un elemento que se diferencia de los demás y cuyas características son determinantes.
* Dentro de cada clase, la información tiene tres niveles de protección : privado, protegido y público.
* La información privada sólo es accesible al código de la misma clase a que pertenece .
* La información protegida es accesible tanto al código de la misma clase como a aquél de clases heredadas.
* La información pública posee el mismo grado de acceso que la protegida, con la diferencia de que puede ser solicitada también por el programa principal.
* Por lo general, el elemento público de una clase es el código, mientras que los datos representan los elementos privados ó protegidos .
* Cada módulo tiene una estructura de clases formada por un nodo y varias raíces. Las clases raíz reúnen información específica al módulo, mientras que las clases nodo ó enlace facilitan la comunicación inter-clases del módulo e inter-módulos .

La figura III.2 de la página siguiente presenta la estructura de clases con sus reglas de herencia .

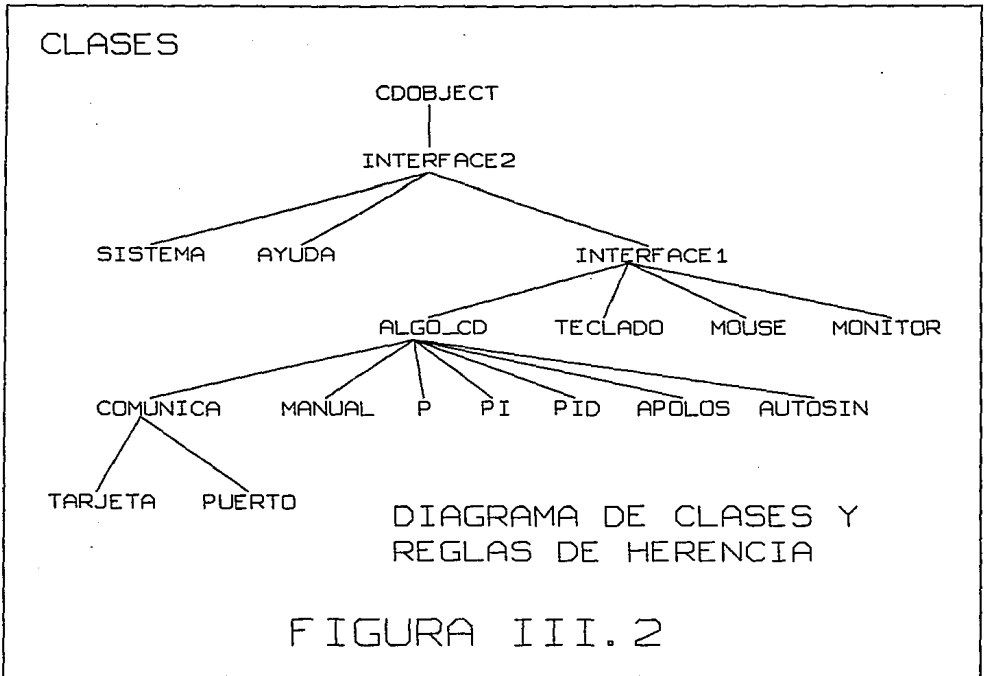

CLASES
CDOBJECT
INTERFACE2
SISTEMA
AYUDA
INTERFACE1
ALGO_CD
TECLADO
MOUSE
MONITOR
COMUNICA
MANUAL
P
PI
PID
APOLOS
AUTOSIN
TARJETA
PUERTO
DIAGRAMA DE CLASES Y
REGLAS DE HERENCIA

FIGURA III.2

b.1) Descripción de Clases .

La descripción de cada clase que forma la estructura de la figura anterior se da a continuación mediante una ficha que posee su nombre, propósito, campos (variables) y operaciones (funciones) . Además de explicar brevemente la tarea de cada campo y operación, se señala, si la hay, la exportación e importación de los elementos de la clase.

Nombre	: Tarjeta .
Propósito	: Contiene código y datos que inicializan la operación de la tarjeta PCL-812, validación de parámetros de conteo y selección de canales, ejecución de conversiones A/D y D/A, así como la deshabilitación de los modos de operación .
Propiedades	: Los campos son protegidos . Las operaciones son públicas . Es clase raíz del módulo de comunicaciones .
Campos :	
(1) int CANADC	: Guarda número de canal A/D .
(2) int CANDAC	: Guarda número de canal D/A .
(3) float FREC	: Guarda frecuencia de muestreo .
(4) float T	: Guarda período de muestreo .
Operaciones :	
(1) Tarjeta	: Inicialización de operación de PCL-812 .
(2) ~Tarjeta	: Deshabilitación de PCL-812 .
(3) validadc	: Validación de canal A/D .
(4) validdac	: Validación de canal D/A .
(5) validfrec	: Validación de frecuencia de muestreo .
(6) counters	: Conteo para período de muestreo .
(7) adconv	: Conversión A/D de valores ± 10 volts .
(8) daconv	: Conversión D/A a valores ± 10 volts .
(9) indat_card	: Ingreso de parámetros de PCL-812 .

Nombre	: Puerto .
Propósito	: Realizar la inicialización, transmisión y recepción de información por el puerto RS-232 de comunicación asíncrona .
Observaciones	: No posee campos ni operaciones pues se desarrollará a futuro . Empero, es clase raíz del módulo de comunicaciones .

Nombre : Comunica .
Propósito : Clase enlace del módulo de comunicaciones para proveer acciones genéricas de E/S para controlador .
Propiedades : El único campo es protegido . Las operaciones son públicas . Hereda elementos públicos de Tarjeta y Puerto.
Campos : (1) int FLAGES : Bandera que indica el uso de tarjeta PCL-812 ó puerto RS-232 . Operaciones : (1) Comunica : Decide que dispositivo de E/S para controlador emplear. (2) in_y : Acción genérica de entrada de señal "Y" . (3) out_u : Acción genérica de salida de señal "U" .

Nombre : Manual .
Propósito : Algoritmo de control manual
Propiedades : Sólo hay operaciones y son públicas . Es clase raiz del módulo de algoritmos de control.
Operaciones : (1) hand_up : Incremento de 0.1 volt a señal de control "U". (2) hand_down : Decremento de 0.1 volt a señal de control . (3) hand_right: Incremento de 1 volt a señal de control . (4) hand_left : Decremento de 1 volt a señal de control . (5) hand_tope : Mantiene a señal de control "U" dentro del rango ± 10 volts .

Nombre : P .
Propósito : Algoritmo de control proporcional P .
Propiedades : Sólo hay una operación y es pública . Es clase raiz del módulo de algoritmos de control.
Operación : (1) algo_p : Ejecuta acción de control proporcional .

Nombre : Pi .
Propósito : Algoritmo de control proporcional-integrativo PI.
Propiedades : Sólo hay dos operaciones y son públicas . Es clase raiz del módulo de algoritmos de control.
Operaciones : (1) e_de_k : Obtiene señal de error e(k) . (2) algo_pi : Ejecuta control proporcional-integrativo .

Nombre : Pid .
Propósito : Algoritmo de control proporcional-integral -derivativo PID estructura (b) .
Propiedades : Los campos son protegidos . Las operaciones son públicas . Es clase raiz del módulo de algoritmos de control .
Campos : (1) double ALFA : Parámetro α de controlador PID . (2) double BETA : Parámetro β de controlador PID . (3) double GAMA : Parámetro γ de controlador PID . (4) double R0 : Coeficiente de y(kT) en ley de control PID. (5) double R1 : Coeficiente de y(kT-T) en ley de control . (6) double R2 : Coeficiente de y(kT-2T) en ley de control. (7) double NUM1 : Guarda suma de términos y(kT) . (8) double NUM2 : Guarda suma de términos r(kT) .
Operaciones : (1) get_alfa : Obtiene parámetro α . (2) get_beta : Obtiene parámetro β . (3) get_gama : Obtiene parámetro γ . (4) get_r0 : Obtiene parámetro R0 . (5) get_r1 : Obtiene parámetro R1 . (6) get_r2 : Obtiene parámetro R2 . (7) get_num1 : Obtiene valor NUM1 . (8) get_num2 : Obtiene valor NUM2 . (9) algo_pid : Ejecuta controlador proporcional-integral -derivativo .

Nombre	: Apolos .
Propósito	: Algoritmo de control para planta de 2 orden obtenido por el método de asignación de polos con cancelación de ceros .
Propiedades	: Los campos son protegidos . Las operaciones son públicas . Es clase raiz del módulo de algoritmos de control .
Campos :	
(1) double P1	: Segundo coeficiente del polinomio Bm(z) .
(2) double P2	: Tercer coeficiente del polinomio Bm(z) .
(3) double S1	: Primer coeficiente del polinomio S(z) .
(4) double S2	: Segundo coeficiente del polinomio S(z) .
(5) double Q1	: Primer coeficiente del polinomio Q(z) .
(6) double Q2	: Segundo coeficiente del polinomio Q(z) .
(7) double T1	: Coeficiente del polinomio T(z) .
Operaciones :	
(1) get_p1_p2	: Obtención de parámetros P1 y P2 .
(2) get_param	: Obtención de S1, S2, Q1, Q2 y T1 .
(3) algo_apolos	: Ejecución de algoritmo de control por asignación de polos .

Nombre	: Autosin .
Propósito	: Corresponde al algoritmo de control autosintonizable .
Observaciones	: No posee campos ni operaciones . Sólo es enunciada . Es clase raiz del módulo de algoritmos de control .

Nombre : Algo_cd .
Propósito : Sirve como clase nodo del módulo de algoritmos de control, enlazándolo a su vez al de comunicaciones . Debido a ésto, sus funciones realizan, junto con la acción de control, la adquisición de datos .
Propiedades : Los campos están protegidos . Las operaciones son públicas . Hereda en forma múltiple las clases Comunica, Manual, P, Pi, Pid y Apolos .
Campos : (1) double U[3] : Arreglo de tres elementos para señal u(kT) (2) double R[3] : Arreglo de tres elementos para señal r(kT) (3) double E[3] : Arreglo de tres elementos para señal e(kT) (4) double Y[3] : Arreglo de tres elementos para señal y(kT) (5) double K : Constante de proporcionalidad . (6) double TI : Tiempo integrativo . (7) double TD : Tiempo derivativo . (8) double N : Constante del filtro paso-bajas del controlador PID . (9) double A : Polo de planta de segundo orden . (10) double B : Cero de planta de segundo orden . (11) double C : Polo de planta de segundo orden . (12) double TR : Tiempo de elevación de respuesta del proceso. (13) double MP : Sobrepaso de la respuesta del proceso . (14) double VY : Valor del voltaje de la variable del proceso . (15) double VU : Valor del voltaje de la señal de control .
Operaciones : (1) Algo_cd : Función de inicialización de parámetros de controladores . (2) vldpar_cd : Validación de parámetros de controladores . (3) indat_con : Ingreso de valores de parámetros .

(4) recorrer : Traslación en el tiempo de señales del controlador .

(5) cd_man : Ejecución de funciones in_y, out_u, recorrer y algoritmo de control manual .

(6) cd_ p : Ejecución de funciones in_y, out_u, recorrer y algoritmo de control proporcional .

(7) cd_pi : Ejecución de funciones in_y, out_u, recorrer y algoritmo de control PI .

(8) pre_pid : Preprocesamiento de controlador PID .

(9) cd_pid : Ejecución de funciones in_y, out_u, recorrer y algoritmo de control PID .

(10) pre_apolos : Preprocesamiento de controlador por asignación de polos .

(11) cd_apolos : Ejecución de funciones in_y, out_u, recorrer y algoritmo de control por asignación de polos.

Nombre : Teclado .
Propósito : Realizar el sensado y control del teclado para la lectura y validación de datos .
Propiedades : Los campos son protegidos . Las operaciones son públicas . Es clase raíz del módulo de Interface .
Campos : (1) *APT : Apuntador a cadena de caracteres . (2) char CAR[80] : Cadena de carateres .
Operaciones : (1) Teclado : Función de inicialización que asigna apuntador a cadena de caracteres . (2) rkeyb : Verifica si se ha presionado una tecla, y en caso afirmativo regresa el valor ASCII de la tecla . (3) invalnum : Función de lectura de valor numérico para ambiente gráfico . (4) invalal : Función de lectura de valor alfanumérico para ambiente gráfico .

Nombre : Monitor .
Propósito : Consta de funciones de inicialización de modo gráfico de alta resolución para monitor VGA; de pantallas de presentación, ejecución y adquisición de datos del usuario; de graficación de señales de control del proceso, etc .
Propiedades : Todos los campos son protegidos . Todas sus operaciones son públicas . Es clase raiz del módulo de Interface .
Campos : (1) int GRAPHDRIVER : Controlador gráfico . (2) int GRAPHMODE : Modo del controlador gráfico . (3) int xpos : Valor del tiempo en pantalla de graficación . (4) int imagenimpar : Bandera que indica si la pantalla de graficación es impar. Esto para efectos de consulta a historia de control del proceso . (5) int DATOSU[882] : Arreglo para valores de señal u(k) . (6) int DATOSY[882] : Arreglo para valores de señal y(k) . (7) int NUMIMA : Número de pantalla de graficación de control de proceso .
Operaciones : (1) Monitor : Inicialización de modo gráfico . (2) ~Monitor : Reestablecimiento de modo previo . (3) box : Traza un cuadro de acuerdo a coordenadas . (4) marco1 : Primer marco de pantalla fuera de línea . (5) marco2 : Segundo marco de pantalla fuera de línea . (6) barra : Menú de pantalla fuera de línea . (7) sismenu : Submenú de la opción de sistema . (8) e_smenu : Submenú de la opción de comunicaciones . (9) conmenu : Submenú de la opción de controladores . (10) fondo1 : Fondo de pantalla fuera de línea . (11) wgen : Ventana de entrada de datos .

(12) fondo2 : Fondo de pantalla en línea .
(13) fondo_barras : Fondo de gráfica de barras .
(14) barras_u_y : Graficación de barras .
(15) pantalla : Pantalla de graficación de control de proceso .
(16) escalay : Escala de amplitud de señales de control de proceso .
(17) escalax : Escala del tiempo para pantalla de graficación de control de proceso .
(18) wmes : Menú de opciones durante operación en línea .
(19) disval : Función polimórfica para el despliegue de valores numéricos, enteros o flotantes .
(20) rotulo : Letrero del paquete .
(21) rotcd : Letrero que indica tipo de controlador digital
(22) wesp : Igual que wgen pero sin ícono "OK" .
(23) ponl : Pantalla para operación en línea .
(24) poffl : Pantalla para operación fuera de línea .
(25) graprintf : Impresión de textos para pantalla en modo gráfico, con número arbitrario de argumentos .
(26) grafica : Graficación de barras y de señales de control
(27) *gimage : Almacenamiento dinámico en memoria de gráficos .
(28) inix : Inicialización de xpos .
(29) baroption : Barra de señalamiento de opción en menús .
(30) put_newscreen : Coloca en pantalla la gráfica actual de control del proceso.
(31) put_lastscreen : Coloca en pantalla la historia de control del proceso .

Nombre : Mouse .

Propósito : Proveer datos y código para el uso del mouse mediante interrupciones al μprocesador . Consta de funciones que inicializan el dispositivo, despliegan y apagan su cursor, indican la posición y estado de los botones del mismo, etc .

Propiedades : Los campos son protegidos .
Las operaciones son públicas .
Es clase raíz del módulo de Interface .

Campos :

(1) union REGS reg : Estructura de dato tipo "union" correspondiente a los registros de los CPU's familia 80x86 .

(2) struct SREGS seg : Estructura de dato tipo "struct" para operación de registros de segmento de CPU's tipo 80x86 .

(3) mstat_t mstat : Estructura de dato tipo "struct" definido por programador para conocimiento del estado del "mouse", con los siguientes campos :

(3.a) mstat.verifica : Bandera que indica si el sistema soporta "mouse" .

(3.b) mstat.modocrt : Indica el modo gráfico/texto del monitor . Util para hacer la conversión de coordenadas físicas del "mouse" a coordenadas de pantalla virtual .

(3.c) mstat.cursoron : Indica estado del cursor del "mouse" .

(3.d) mstat.botones : Indica que botón del "mouse" se ha presionado .

(3.e) mstat.numpress: Indica si se he presionado algún botón .

(3.f) mstat.x : Posición en "x" del "mouse" en pantalla

(3.g) mstat.y : Posición en "y" del "mouse" en pantalla

(3.h) mstat.minx : Límite mínimo en "x" para el "mouse" en pantalla .

3.i) mstat.maxx : Límite máximo en "x" para el "mouse" en pantalla .

(3.j) mstat.miny : Límite mínimo en "y" en pantalla .

(3.k) mstat.maxy : Límite máximo en "y" en pantalla .

Operaciones :

(1) reset : Haciendo uso de la interrupciones 0x10 y 0x33, manejadas por BIOS y DOS respectivamente, averigua el modo CRT y si hay driver del mouse.

(2) Mouse : Función constructor que inicializa la operación del "mouse" obteniendo la dirección de la rutina de servicio a la interrupción 0x33 y ejecutando la función reset .

(3) ~Mouse : Función destructor que deshabilita uso del "mouse".

(4) initok : Regresa la bandera verifica, de la estructura mstat .

(5) showcursor : Despliega cursor del "mouse" .

(6) hidecursor : Apaga cursor .

(7) getposition : Regresa dirección de la estructura mstat habiendo almacenado en ella la posición y estado de los botones al ser presionado uno de ellos y mantenerse así .

(8) setposition : Asigna posición del cursor del "mouse".

(9) getbutton : Regresa dirección de la estructura mstat habiendo averiguado el estado de uno sólo de los botones y la posición del cursor .

(10) zonax : Define límites de movimiento para el cursor del "mouse" en "x" .

(11) zonay : Igual que en zonax pero para "y" .

Nombre : Interface1 .
Propósito : Servir como una de las dos clases nodo del módulo interface sistema-usuario . Consta de funciones que presentan ventanas de entrada de datos para parámetros de controladores y de comunicaciones .
Propiedades : No hay campos . Las operaciones son públicas . Hereda elementos públicos de las clases Algo_cd, Teclado, Monitor y Mouse .
Operaciones : (1) we_s : Ventana para parámetros para la comunicación del controlador . (2) wcon : Ventana para parámetros de controladores . (3) wonl : Ventana para entrada de datos durante la operación en línea . (4) aonl : Lectura de valores numéricos de teclado durante la operación en línea . (5) nodisp : Ventana con mensaje para opciones no disponibles . (6) dgraph : Despliega gráfica de control de proceso .

Nombre : Sistema .

Propósito : Proporcionar funciones para apoyar al sistema .

Propiedades : Es única clase raíz del módulo de apoyo al sistema . Las operaciones son públicas. Los campos son protegidos .

Campos :

(1) struct linked_list : Estructura de datos usada para crear una lista ligada cuyos elementos contengan información de archivos del directorio corriente Sus campos son :

- (1.a) char nombre[13] : Cadena de caracteres para el nombre del archivo .
- (1.b) long space : Tamaño del archivo .
- (1.c) struct lista *next : Apuntador a siguiente elemento de lista .

(2) linked_list *ORIGEN : Apuntador a cabeza de lista ligada formada con elementos linked_list .

(3) linked_list *MOVIL : Apuntador móvil a elementos de lista ligada linked_list .

Operaciones :

(1) verunit : Verifica si las unidades de disco flexible o duro están listas .

(2) dirlist : Mediante asignación dinámica de la memoria, revisa el directorio corriente para hallar subdirectorios, formando con ésta información elementos de una lista ligada .

(3) filelist : Empleando asignación dinámica de la memoria ejecuta lo mismo que dirlist pero para archivos .

(4) dlist : Regresa la dirección de inicio de la lista ligada que contiene información de subdirectorios y archivos del directorio corriente .

(5) libmemdir : Libera la memoria empleada para guardar la lista ligada con información del directorio.

(6) cambiardir : Realiza cambio de directorio si es posible. En caso negativo lo notifica .

(7) copiar : Hace el copiado de archivos si es posible. En caso negativo lo notifica .

(8) fechador : Entrega la dirección de una cadena de caracteres con información de la hora y fecha del sistema .

Nombre : Ayuda .
Propósito : Proporcionar apoyo al usuario mediante un archivo que contiene información para el manejo del paquete .
Propiedades : Es única clase raiz del módulo de apoyo al usuario . Los campos son protegidos . Las operaciones son públicas .
Campos : (1) char *PTR : Apuntador tipo caracter para la lectura de archivo de ayuda . (2) char *PTRS[200] : Arreglo de apuntadores a cadenas de caracteres para su despliegue en pantalla . (3) int LIMITE : Número límite de renglones de archivo de ayuda . (4) int POS : Indica el número del renglón del archivo de ayuda en el que se encuentra el usuario para su lectura .
Operaciones : (1) memf : Asigna espacio de memoria en forma dinámica al archivo de ayuda .. Si no hay lo notifica . (2) rfile : Lee el archivo de ayuda de disco, colocando la información en memoria . (3) chptrs : Cambia la posición del renglón para su lectura (4) libptr : Libera la memoria asignada al archivo de ayuda

Nombre	: Interface2 .
Propósito	: Servir como la segunda clase nodo del módulo de interface sistema-usuario, ligando a las clases Sistema y Ayuda con Interface1. Implementa ventanas para el ingreso de datos de la opción Sistema y para la consulta de Ayuda .
Propiedades	: No posee campos . Las operaciones son públicas. Hereda los elementos públicos de las clases Sistema, Ayuda e Interface1.
Operaciones	:
(1) wdir	: Ventana de despliegue del directorio actual .
(2) wchdir	: Ventana para cambio de directorio .
(3) wcopy	: Ventana para copiado de archivos .
(4) wtime	: Ventana para despliegue de hora y fecha del sistema .
(5) whelp	: Ventana para despliegue de archivo de ayuda para el manejo del paquete .

Nombre : Cdobject .

Propósito : Realizar la operación en y fuera de línea .

Propiedades : Su único campo es protegido .
Las operaciones son públicas .
Hereda Interfacei y es clase terminal .

Campos :

(1) int CDTYPE : Indica que tipo de controlador digital se está empleando .

Operaciones :

(1) Cdobject : Constructor que hace que CDTYPE no tenga valor útil alguno .
(2) op_offline : Ejecuta la operación fuera de línea .
(3) op_cdman : Operación de controlador manual .
(4) op_cdp : Operación de controlador proporcional .
(5) op_cdpi : Operación de controlador PI .
(6) op_cdpid : Operación de controlador PID .
(7) op_cdapolos : Operación de controlador por asignación de polos .
(8) op_online : Ejecuta la operación en línea de acuerdo con el controlador digital escogido de acuerdo al valor de CDTYPE .

b.2) Diagramas de flujo .

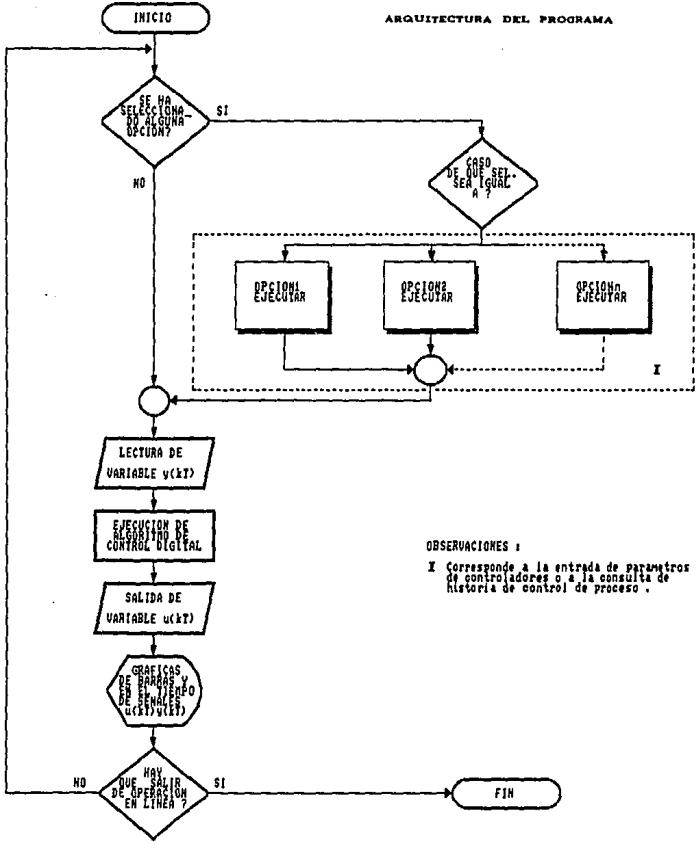
El funcionamiento del programa puede explicarse en forma sintetizada mediante la presentación de diagramas de flujo.

Para la exposición de la operación en línea, los diagramas de flujo mostrados en la presente sección comprenden :

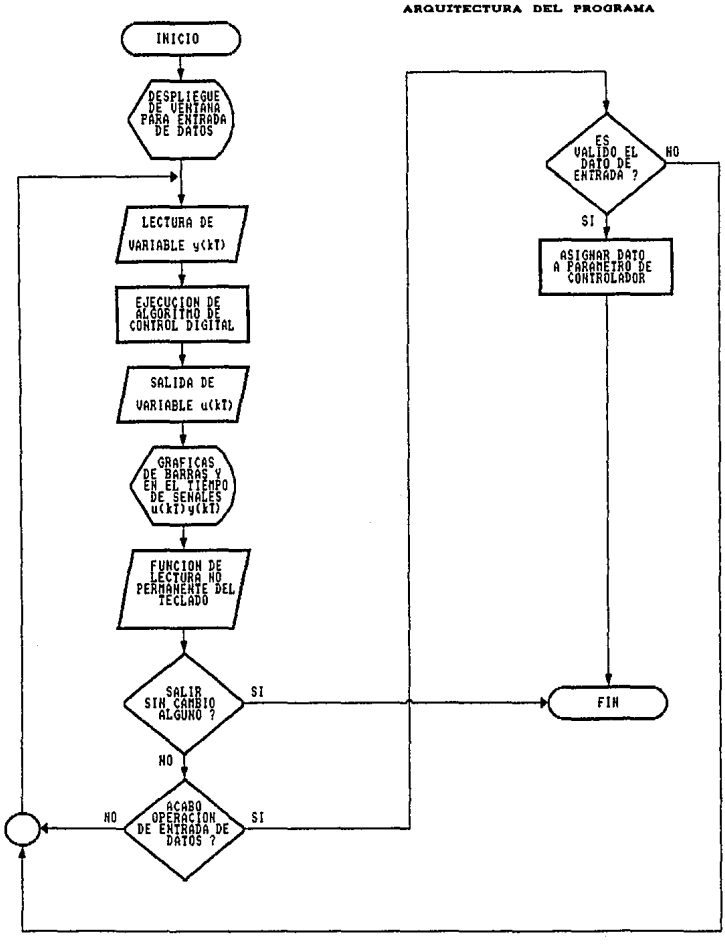
DF.1) Generalización de operación en línea .
DF.1.a) Entrada de datos durante operación en línea .
DF.1.b) Consulta de historia de control de proceso en línea .
DF.2.a) Inicialización de operación de tarjeta PCL-812 .
DF.2.b) Lectura A/D de tarjeta PCL-812 .
DF.2.c) Salida D/A de tarjeta PCL-812 .
DF.3.a) Algoritmo de control digital Manual .
DF.3.b) Algoritmo de control digital Proporcional P .
DF.3.c) Algoritmo de control digital Proporcional-integrativo PI .
DF.3.d) Preprocesamiento del controlador digital PID-b .
DF.3.e) Algoritmo de control digital Proporcional-integrativo-derivativo PID-b .
DF.3.f) Preprocesamiento del controlador por asignación de polos con cancelación de ceros .
DF.3.g) Algoritmo de control digital por asignación de polos con cancelación de ceros para planta de 2º orden .

De la misma forma, para la operación fuera de línea los diagramas de flujo son :

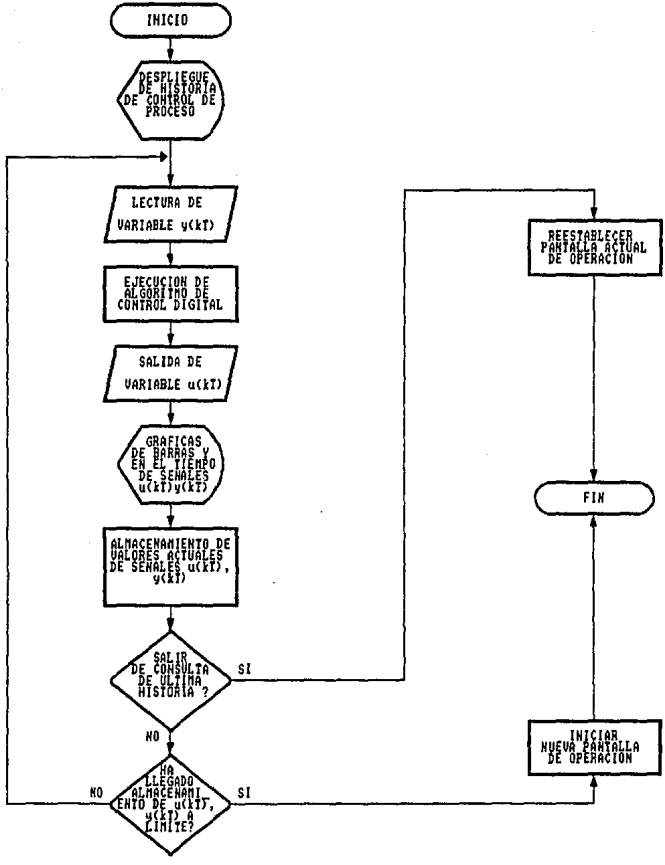
DF.4) Generalización de operación fuera de línea .
DF.5) Lectura no permanente de teclado para entrada de datos.
DF.6.a) Primera parte de ventana para entrada de datos .
DF.6.b) Segunda parte de ventana para entrada de datos .



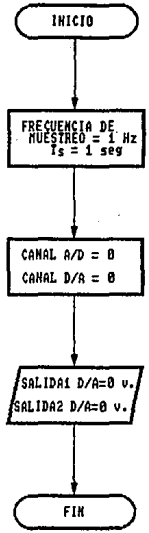
DF.1 GENERALIZACION DE OPERACION EN LINEA .



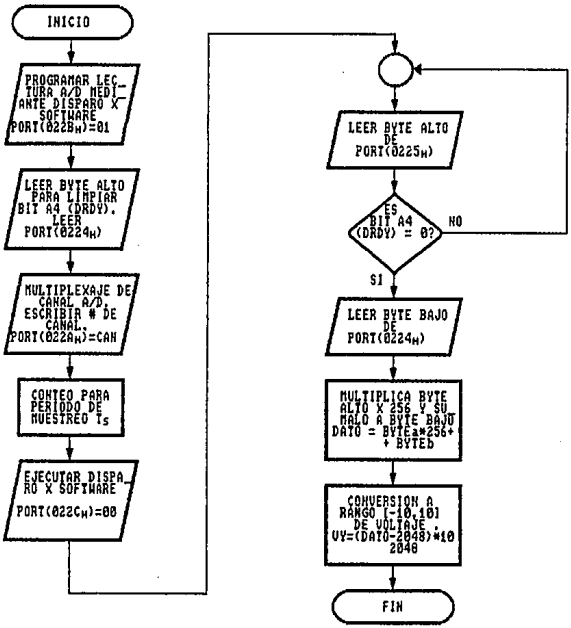
DF.1.a MODIFICACION DE PARAMETRO DEL CONTROLADOR DIGITAL DURANTE OPERACION EN LINEA . 55



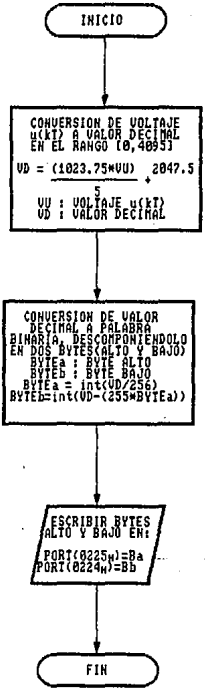
DF.1.b CONSULTA DE ULTIMA HISTORIA DE CONTROL DE PROCESO .



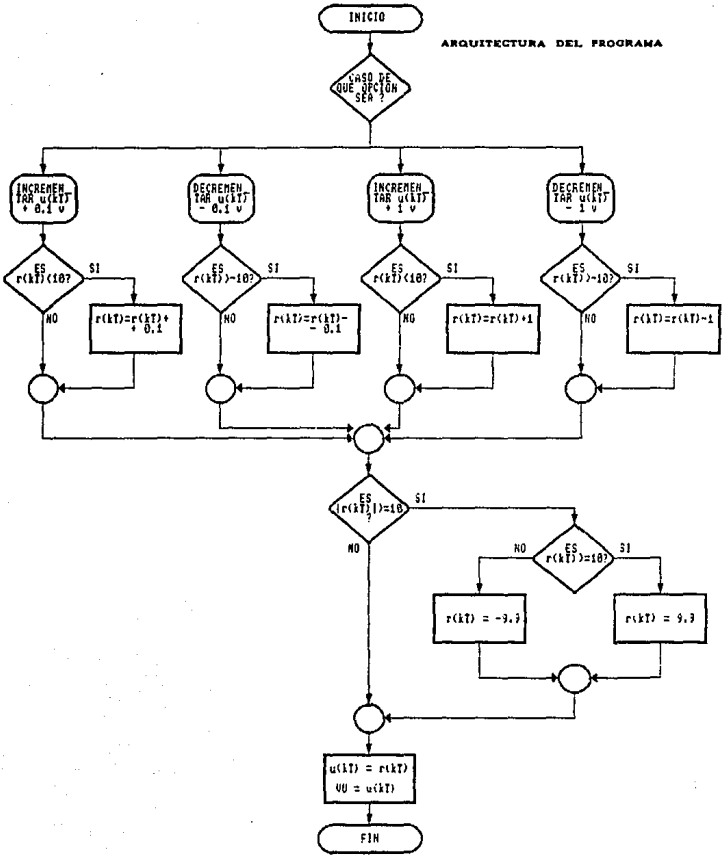
DF.2.a INICIALIZACION DE OPERACION DE TARJETA PCL-812 .



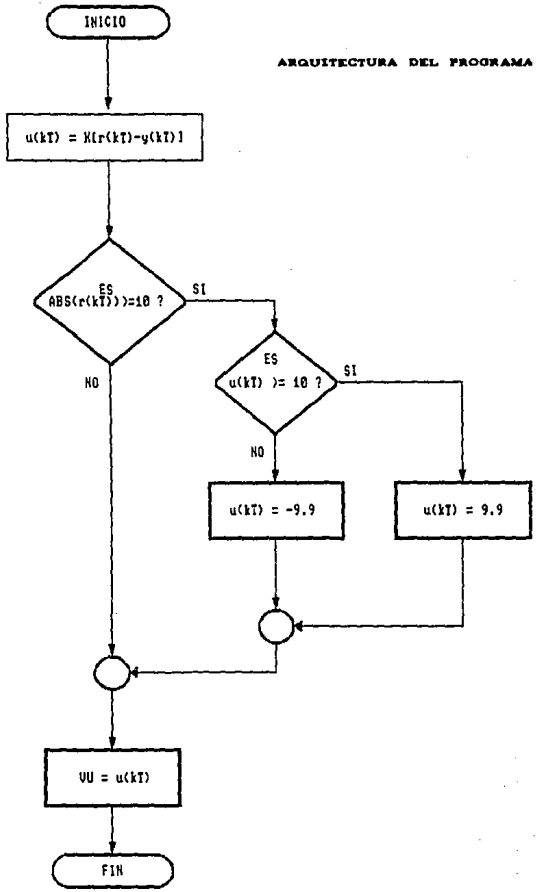
DF.2.b LECTURA A/D DE TARJETA PCL-812 .



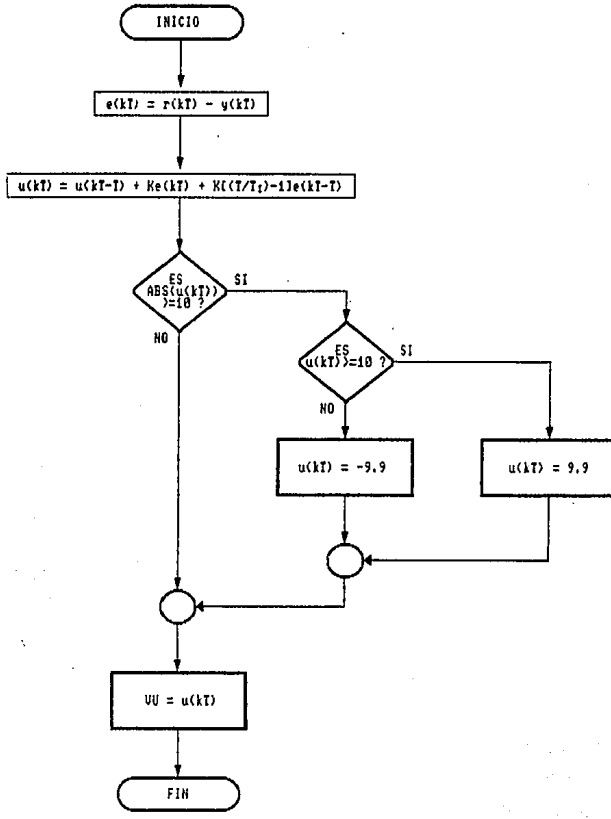
DF.2.c SALIDA D/A POR CANAL1 DE TARJETA PCL-812 .



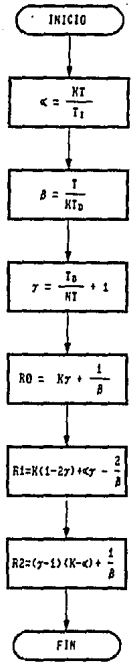
DF.3.a ALGORITMO DE CONTROL MANUAL .



DF.3.b ALGORITMO DE CONTROL PROPORCIONAL P .



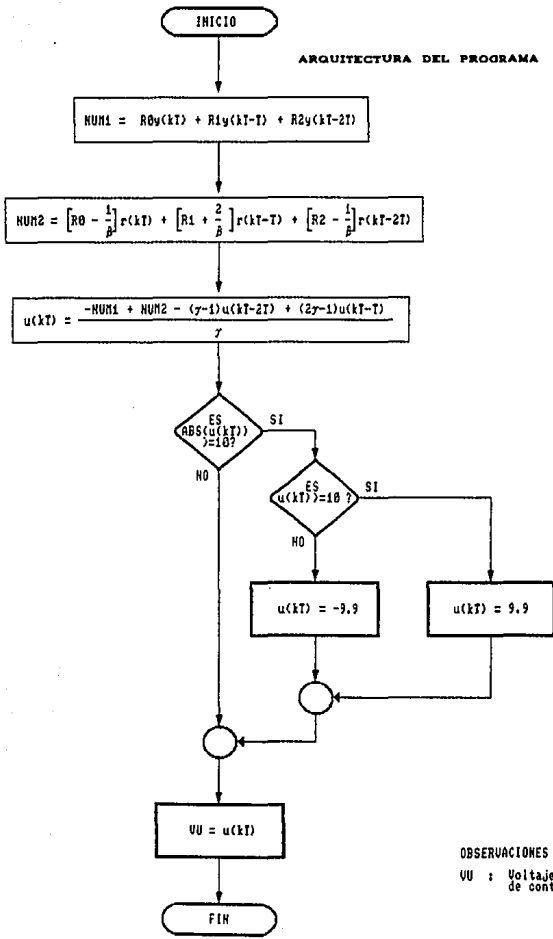
DF.3.c ALGORITMO DE CONTROL PROPORCIONAL-INTEGRATIVO PI .



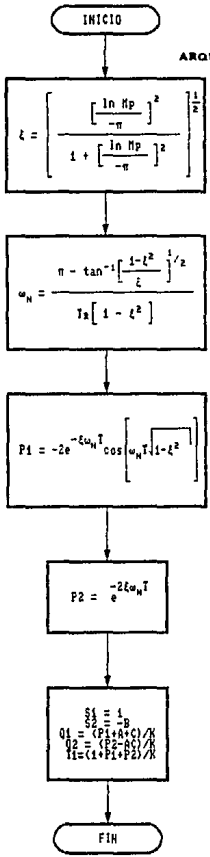
OBSERVACIONES :

K : Constante de proporcionalidad .
T : Periodo de muestreo
T_I : Tiempo integrativo
T_D : Tiempo derivativo
N : Constante del filtro paso-bajas .

DF.3.d PREPROCESAMIENTO DEL CONTROLADOR PROPORCIONAL-INTEGRATIVO-DERIVATIVO PID .



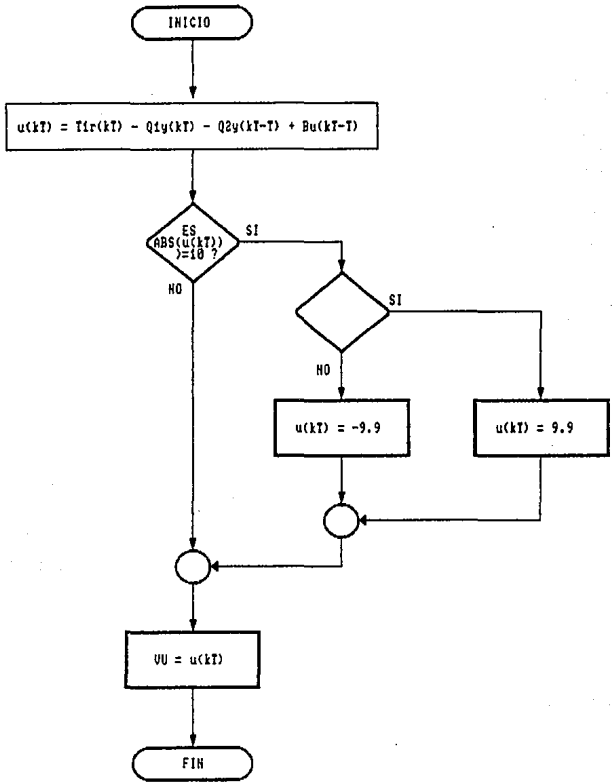
DF.3.e ALGORITMO DE CONTROL PROPORCIONAL-INTEGRATIVO-DERIVATIVO PID .



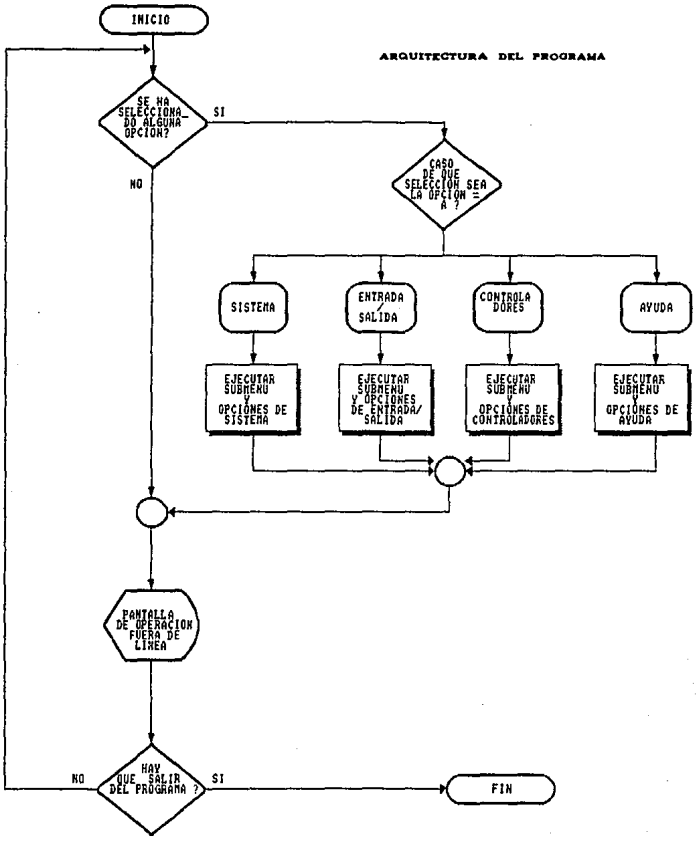
OBSERVACIONES :

Mp : Sobrepaso .
Ta : Tiempo de elevacion.
ξ : Factor de amortiguamiento .
ω_N : Frecuencia natural
P1,P2 : Coeficientes del polinomio A(z).
S1,S2 : Coeficientes del polinomio S(z) .
Q1,Q2 : Coeficientes del polinomio Q(z) .
T1 : Coeficiente del polinomio T(z) .

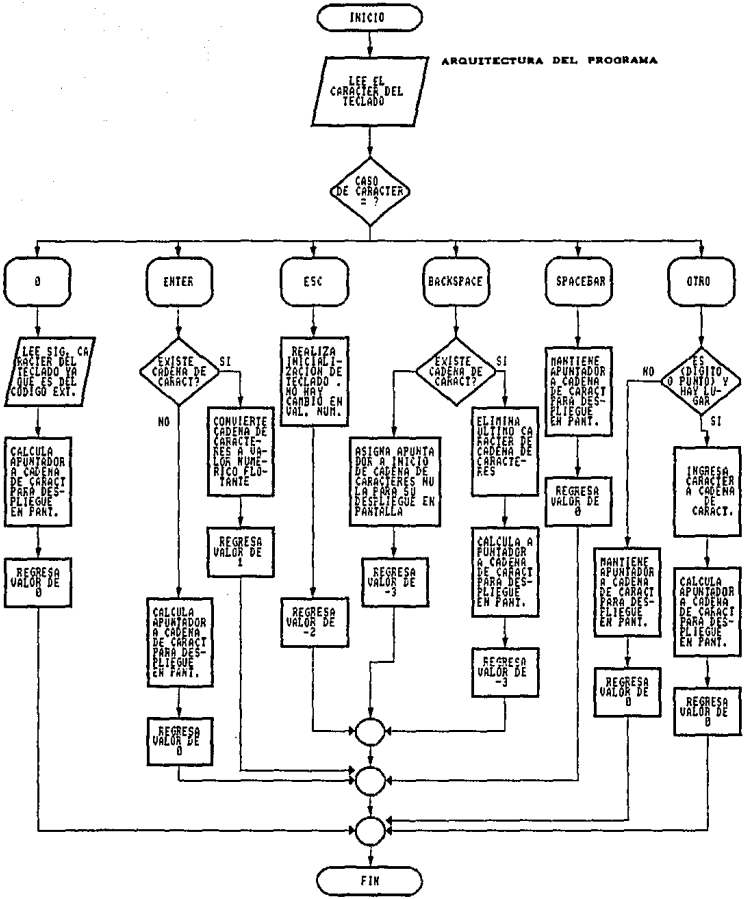
DF.3.f PREPROCESAMIENTO DEL CONTROLADOR POR ASIGNACION DE POLOS CON CANCELACION DE CEROS PARA PLANTA DE 2° ORDEN .



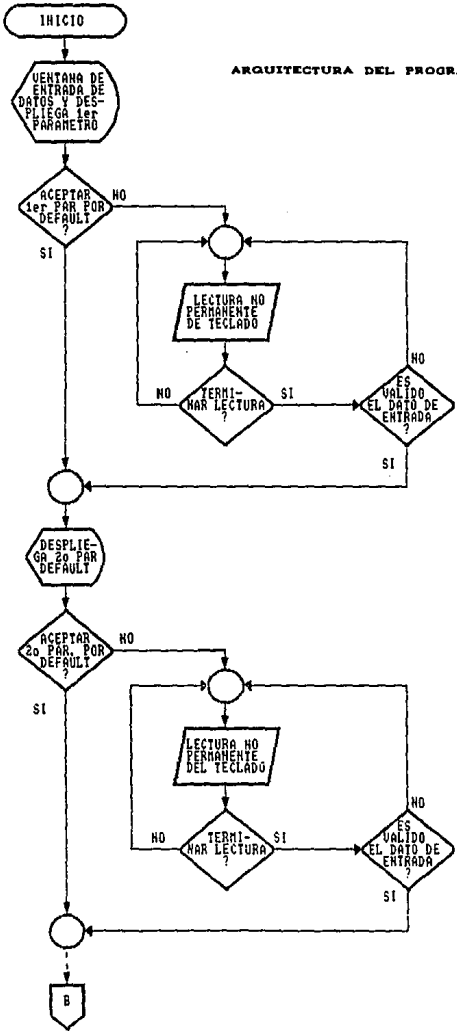
DF.3.g ALGORITMO DE CONTROL POR ASIGNACION DE POLOS CON CANCELACION DE CEROS PARA PROCESO DE 2º ORDEN .



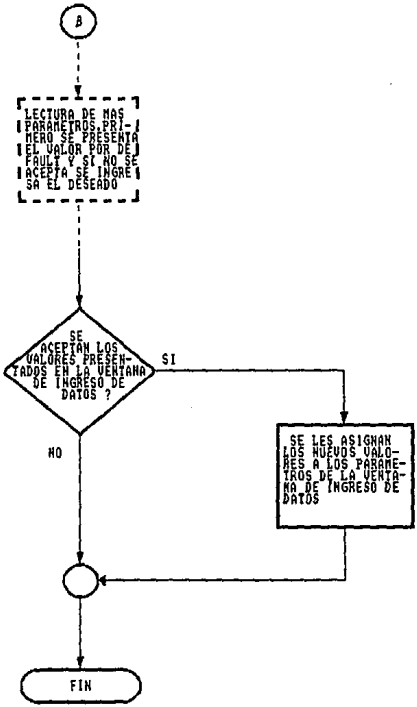
DF.4 GENERALIZACION DE OPERACION FUERA DE LINEA .



DF.5 LECTURA NO PERMANENTE DE TECLADO PARA ENTRADA DE DATOS NUMERICOS .



DF.6.A GENERALIZACION DE VENTANA PARA ENTRADA DE DATOS .



DF.6.B GENERALIZACION DE VENTANA PARA ENTRADA DE DATOS .

CAPITULO IV

PRUEBAS DEL PAQUETE

a) Pruebas funcionales :

Son aquellas que a través de entradas nominales se esperan obtener ciertos resultados. Como el objetivo del paquete es ejecutar acciones de control digital sobre un proceso, es de interés principal que las pruebas funcionales se realizen bajo éste tipo de operación, siendo sólo objeto de análisis somero la respuesta transitoria del sistema.

Para simular el comportamiento de una planta de 2^{o} orden se empleó el simulador de procesos G26 con la siguiente función de transferencia :

$$Gp(s) = \frac{1}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

donde : τ_n Corresponde a un valor de tiempo continuo, siendo $\tau_1 = 1$ seg. y $\tau_2 = 0.7$ seg.

quedando $Gp(s)$ finalmente como :

$$Gp(s) = \frac{1}{(s + 1)(0.7s + 1)}$$

Los controladores empleados durante las pruebas fueron :

* Controlador proporcional-integrativo PI.
* Controlador proporcional-integrativo-derivativo-b PID-b
* Por asignación de polos con cancelación de ceros.

Para todos los controladores se empleó la misma frecuencia de muestreo : 10 Hz .

a.1) Prueba con controlador PI .

Los valores dados a sus parámetros fueron :

$$K = 2$$
$$T_i = 0.5$$
$$REF = 4.0$$

la figura IV.1 presenta la respuesta del sistema. Como se puede ver se trata de un sistema subamortiguado.

a.2) Prueba con controlador PID-b .

Los valores dados a sus parámetros fueron :

$$K = 2$$
$$T_i = 0.5$$
$$T_d = 0.5$$
$$N = 10$$
$$REF = 4.0$$

la figura IV.2 muestra la respuesta del sistema.

a.3) Prueba con controlador por asignación de polos con cancelación de ceros.

Los valores dados a sus parámetros fueron :

$$K = 0.0065909$$
$$M_p = 0.15$$
$$T_r = 0.5$$
$$a = 0.80483$$
$$b = -0.921983$$
$$c = 0.86688$$
$$REF = 4.0$$

la respuesta de la planta al cambio de referencia citado se muestra en la figura IV.3 .

Los valores de los parámetros K, a, b, y c que representan la constante de proporcionalidad, el cero y los polos de la función de transferencia discreta se obtuvieron mediante la técnica de Retén de Orden Cero aplicada a la funcion de transferencia continua $G_p(s)$.

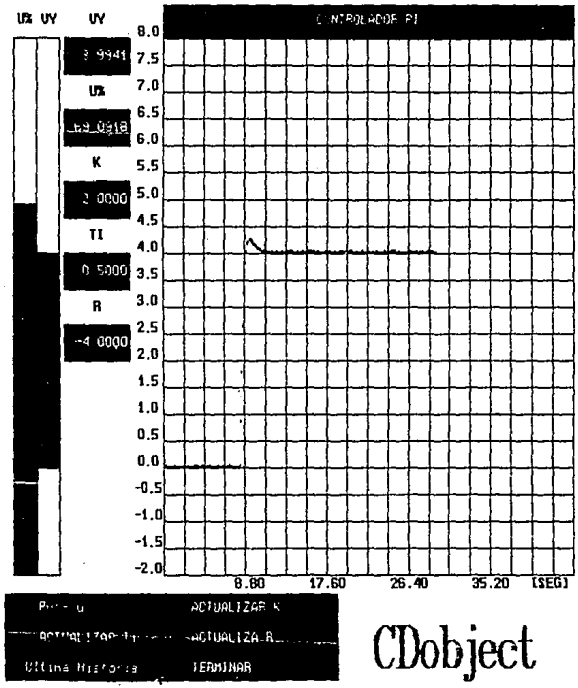


FIGURA IV.1 Prueba del controlador PI .

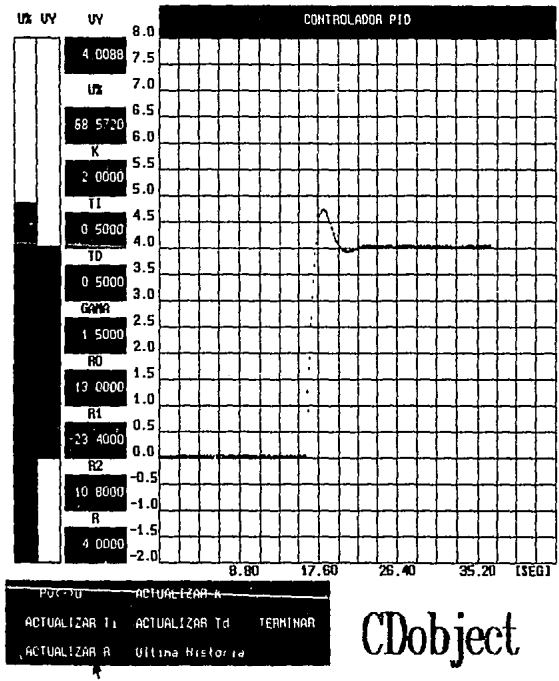


FIGURA IV.2 Prueba del controlador PID-b

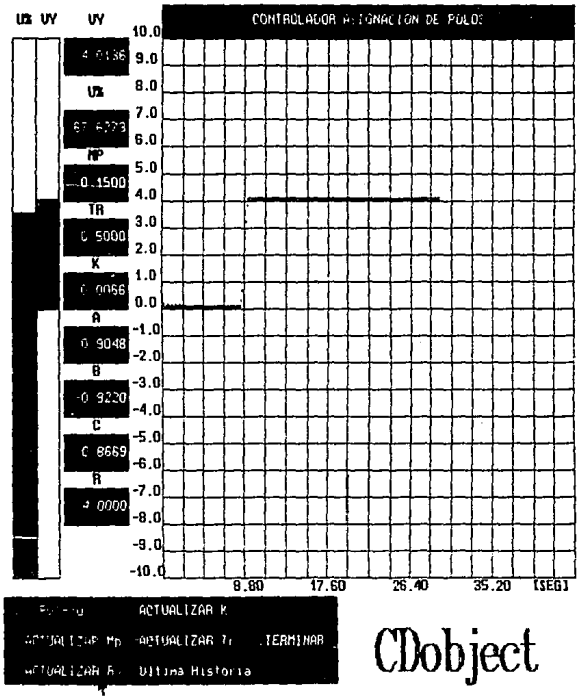


FIGURA IV.3 Prueba del controlador por asignación de polos .

b) Pruebas de tensión :

Son a las que se somete el paquete con el fin de, a través de esfuerzos intencionales,desestabilizar su funcionamiento. Es decir, se prueba el programa bajo aquellas condiciones que pueden inducir al error o a un mal funcionamiento.

La presencia de errores durante éste tipo de pruebas pueden deberse a uno o varios de los siguientes factores :

a) Errores de programación .
b) Condiciones de compilación y ligado .
c) Uso indebido del programa por parte del usuario .
d) Fallas del Hardware .

De las tres causas anteriores, la tercera es la mayoría de las ocasiones la más difícil de eliminar durante el período de programación, aunque se prevean situaciones y por supuesto sus soluciones .

Durante el período de pruebas y depuración se sometió al programa a situaciones de :

(1) Entrada de datos indebidos .
(2) Condiciones erróneas del sistema .

En el primer caso anterior, se dotó al paquete de código de validación para la entrada de parámetros de E/S , el controlador mismo, operaciones del disco, etc. En el segundo caso se hizo lo mismo pero para detección de cierto equipo de hardware : mouse y unidades de disco .

Mención aparte merecen las fallas de hardware. Un ejemplo de éstas lo es la sucedida en una PC 286 marca DELL

del Departamento de Ingeniería de Control. Durante la operación en línea el desplazamiento del mouse provocaba el despliegue de caracteres ASCII en las gráficas de las señales de control. Al mismo tiempo se sometió el paquete a pruebas en otras dos máquinas 386 del Departamento y de la misma marca así como en otras dos, de modelo 286 y 486 respectivamente. En ninguna de éstas últimas cuatro PC's se observó la falla mencionada. Mediante un programa de diagnóstico de Hardware/Software pudo encontrarse que el equipo que presentaba la anomalía posee un error en el ROM-BIOS del sistema. Este error afecta la rutina de atención a la interrupción de video 0x10, la cual es utilizada por el controlador del mouse del programa para conocer el modo de video del monitor y así poder emplear el cursor adecuado del mouse. Como durante la operación en línea este cursor permanece restringido a un área, no se apaga ni se enciende cada vez que se hace una operación de video, lo cual provoca que la falla de despliegue se haga aparente pues la subrutina no reportó con exactitud al controlador del programa el modo de video en el cual se encuentra el monitor.

La siguiente lista presenta los mensajes de error debidos a ciertas condiciones de prueba de tensión.

Durante la inicialización :

MENSAJE :
Error del sistema gráfico : Device driver file not found (*.BGI) .
CAUSA :
Falta archivo.BGI en directorio de trabajo.
CONSECUENCIA :
Ejecución interrumpida .

MENSAJE :
Error del sistema gráfico : Font file not found (*.CHR)
CAUSA :
Falta archivo.CHR en directorio de trabajo.
CONSECUENCIA :
Ejecución interrumpida .
MENSAJE :
Error : Mouse no instalado en sistema .
CAUSA :
Hardware, ó software del fabricante del mouse no se encuentra instalado en el sistema.
CONSECUENCIA :
Ejecución interrumpida .

Durante la ejecución del programa fuera de línea :

Opción Dir :
MENSAJE :
Error : Unidad de disco no lista. Apriete ESC para seguir.
CAUSA :
No hay disco en la unidad requerida para visualización de directorio.
CONSECUENCIA :
No se puede ver el directorio. No afecta en ninguna otra forma al funcionamiento del paquete.

Opción Cambiar dir :
MENSAJE :
Error : Trayecto no válido. Apriete ESC para seguir.
CAUSA :
El trayecto para cambio de directorio es incorrecto.
CONSECUENCIA :
Ninguna, no afecta el funcionamiento del programa.

MENSAJE :

Error : Directorio no encontrado. Apriete ESC para seguir.

CAUSA :

El directorio a cambiar dado en el trayecto no existe en la unidad de disco de trabajo actual.

CONSECUENCIA :

Ninguna, no afecta el funcionamiento del programa.

MENSAJE :

Error : unidad de disco no lista. Apriete ESC para seguir.

CAUSA :

La unidad de disco (flexible o duro) dada en el trayecto de búsqueda y cambio de directorio no está disponible.

CONSECUENCIA :

Ninguna, no afecta el funcionamiento del programa.

Opción cOpiar :

MENSAJE :

Error : Unidad de disco no lista. Apriete ESC para seguir.

CAUSA :

Una de las unidades de disco dadas en los trayectos origen y destino no está lista.

CONSECUENCIA:

No se realiza el copiado de archivos. Ninguna otra que afecte el funcionamiento del programa.

MENSAJE :

Error : Trayecto no válido. Apriete ESC para seguir.

CAUSA :

Alguno de los dos trayectos es incorrecto.

CONSECUENCIA :

No se realiza el copiado de archivos. Ninguna otra que

afecte el funcionamiento del programa.

MENSAJE :
Error : de escritura . Apriete ESC para seguir .
CAUSA :
Posiblemente el disco destino está lleno.
CONSECUENCIA :
Copiado imperfecto. Ninguna otra que afecte el funcionamiento del programa.

MENSAJE :
Error : de lectura . Apriete ESC para seguir .
CAUSA :
Lectura de archivo incorrecta .
CONSECUENCIA :
Copiado imperfecto. Ninguna otra que afecte el funcionamiento del programa.

MENSAJE :
Error : protección de escritura . Apriete ESC para seguir.
CAUSA :
El disco destino posee protección contra escritura.
CONSECUENCIA :
No se realiza el copiado de archivos. Ninguna otra que afecte el funcionamiento del programa.

Opción Tarjeta PCL-812 :
MENSAJE :
Sonido de frecuencia de 1Khz durante 1 seg.
CAUSA :
Ingreso de parámetros fuera de rango.
CONSECUENCIA :
Se requiere de nuevo el ingreso del dato. Ninguna otra que afecte el funcionamiento del programa.

Opción P :
MENSAJE :
Sonido de frecuencia de 1 Khz durante 1 seg.
CAUSA :
Ingreso de parámetros fuera de rango
CONSECUENCIA :
Se requiere de nuevo el ingreso del dato. Ninguna otra que afecte el funcionamiento del programa.

Opción pI :
MENSAJE :
Sonido de frecuencia de 1 Khz durante 1 seg.
CAUSA :
Ingreso de parámetros fuera de rango
CONSECUENCIA :
Se requiere de nuevo el ingreso del dato. Ninguna otra que afecte el funcionamiento del programa.

Opción piD :
MENSAJE :
Sonido de frecuencia de 1 Khz durante 1 seg.
CAUSA :
Ingreso de parámetros fuera de rango
CONSECUENCIA :
Se requiere de nuevo el ingreso del dato. Ninguna otra que afecte el funcionamiento del programa.

Opción Apolos :
MENSAJE :
Sonido de frecuencia de 1 Khz durante 1 seg.
CAUSA :
Ingreso de parámetros fuera de rango
CONSECUENCIA :
Se requiere de nuevo el ingreso del dato. Ninguna otra que afecte el funcionamiento del programa.

Opción Rango <Pv> :
MENSAJE :
Sonido de frecuencia de 1 Khz durante 1 seg.
CAUSA :
Ingreso de parámetros fuera de rango .
CONSECUENCIA :
Se requiere de nuevo el ingreso del dato. Ninguna otra que afecte el programa.
Opción Correr... :
MENSAJE :
Tiene que escoger primero el controlador . Apriete ESC para seguir .
CAUSA :
No se ha seleccionado controlador alguno .
CONSECUENCIA :
Ninguna que afecte el funcionamiento del programa .

Opción Ayuda :
MENSAJE :
Error : CDOHELP no disponible o espacio insuficiente en memoria . Apriete ESC para seguir .
CAUSA :
En el directorio de trabajo actual no se encuentra el archivo CDOHELP o no hay espacio en el heap .
CONSECUENCIA :
No se puede ver la ayuda al usuario.

Durante la ejecución del programa en línea :

MENSAJE :
La ventana de ingreso de datos se limpia y permanece al ingresar un nuevo valor de algún parámetro del controlador.

CAUSA :

El valor se encuentra fuera de rango.

CONSECUENCIA :

La ventana permanecerá hasta que se ingrese un valor correcto o se abandone la opción presionando ESC.

CAPITULO V

MANUAL DEL USUARIO

a) Introducción :

En prácticas y proyectos del Laboratorio de control digital de la Facultad de Ingeniería se han venido utilizando paquetes de programación que realizan acciones de control digital sobre procesos continuos. A pesar de haber sido desarrollados en un lenguaje estructurado, PASCAL, han presentado deficiencias, no sólo en su ejecución sino también en su estructuración; lo cual ha impedido en forma determinante su mantenimiento y posterior evolución, es decir, la posible incorporación de nuevas técnicas de control y la utilización de código del programa por parte de otras aplicaciones desarrolladas con los mismos fines.

Por tal motivo se pensó en la creación de un nuevo paquete de controladores digitales que proveyera una gran confiabilidad en la ejecución así como un alto grado de estructuración que permitiera remediar las deficiencias de programación ya mencionadas.

Codificado en lenguaje C++, mediante la programación orientada a objetos, CDobject es el nuevo paquete de controladores digitales que opera en el clásico esquema de lazo cerrado con retroalimentación negativa unitaria. Debido a este esquema de control, el ambiente y equipo de ejecución (DOS y PC's respectivamente), sus fines son puramente académicos ; mostrando , sin embargo, algunas características propias del software de control desarrollado para utilización comercial.

La adquisición de datos para el controlador se realiza mediante conversiones A/D y D/A ejecutadas por la tarjeta PCL-812. Con una resolución de 12 bits para ambos tipos de

conversión, ésta tarjeta lee la señal de la planta en un rango de -10 a 10 volts y entrega una señal de control dentro de un rango de 0 a 5 volts, siendo ampliado a uno similar al de la entrada a través de una circuitería especial.

Los controladores digitales disponibles son :

* Manual : La referencia es dada manualmente por el operador y representa la señal de control del proceso. Es usado principalmente para la verificación del alambrado de la planta con la computadora.

* Proporcional : La señal de control es igual a la señal de error amplificada por una constante de proporcionalidad. Debido a que la representación continua de éste controlador no utiliza elementos dinámicos, el equivalente digital es el mismo.

* Proporcional-Integrativo : Corresponde a la aproximación rectangular en adelanto del PI continuo. Además de la constante de proporcionalidad posee otra de tiempo integrativo.

* Proporcional-Integrativo-Derivativo : Obtenido mediante la suma del proporcional-integrativo digital y la aproximación rectangular en atraso del derivativo con filtro paso-bajas. Junto con los parámetros del PI tiene las constantes de tiempo derivativo y el número de componentes o muestras por periodo de oscilación de la señal, es decir, la relación existente entre la frecuencia de muestreo y el ancho de banda de la señal de oscilación amortiguada.

* Por asignación de polos con cancelación de ceros : Ley de control obtenida en base a un conjunto de reglas mediante el conocimiento de la función de transferencia de la planta Gp(z) (en éste caso de segundo orden) y de especificaciones en términos del comportamiento dinámico deseado en malla cerrada.

El paquete tiene dos tipos de operación : en y fuera de línea. Fuera de línea implica que no se ejerce acción alguna de control. En línea significa que la computadora actúa como controlador digital y despliega en pantalla tanto la señal de la planta como la de control. Básicamente se utilizan dos tipos de gráficas para éste propósito. Una de ellas es la gráfica de barras, la cual muestra la amplitud de las señales de interés. Otra es la de graficación en el tiempo que presenta las amplitudes y el comportamiento dinámico del sistema. Por otro lado, dentro de la operación en línea y para cada tipo de controlador existen opciones que permiten la alteración de parámetros de aquéllos y la observación de una breve historia de control del proceso, sin interrumpirse la acción controladora en ningún momento.

Bajo ambos tipos de operación el programa provee una interface sistema-usuario muy amigable. Mediante el uso frecuente de menús, submenús, ventanas de presentación e ingreso de datos además de utilizar mouse, el programa facilita su propio manejo, así como la rápida comprensión de la información presentada, debido a una clara visualización. Esto es importante en cualquier momento de la ejecución del paquete.

Fuera de línea pueden seleccionarse los parámetros necesarios para el manejo de la tarjeta de adquisición de datos y los controladores. También puede hacerse uso de las

opciones de apoyo al sistema y al usuario. Por otra parte, las condiciones iniciales del sistema al momento del arranque son nulas. En caso de abandonar un controlador para utilizar otro las condiciones del sistema se mantienen en su último valor.

En apoyo al sistema pueden visualizarse directorios de las unidades de disco, copiarse archivos de una unidad a otra, cambiar de directorio de trabajo, conocer la hora y fecha del sistema, etc.

Para apoyar al usuario existe una ventana de ayuda en la cual se presenta información útil para el manejo del programa, qué opciones presenta, como ingresar datos, etc.

b) Corrida de muestra :

La instalación del paquete puede darse en disco flexible o duro mediante la ejecución del programa "INSTALAR", y los requerimientos para ella se notifican durante la misma.

Para ejecutar el paquete basta con posicionarse en el directorio de trabajo "CDOBJECT" de la unidad de disco seleccionada durante la instalación y teclear CDOBJECT . De entrada la operación del paquete es fuera de línea y se tiene un menú principal con cuatro opciones :

* Sistema .
* Entrada/salida .
* Controladores .
* Ayuda .

Tanto Sistema como Ayuda presentan opciones cuya utilización no interfiere en forma alguna con la operación de la computadora como controlador. Entrada/salida y Controladores en cambio muestran las distintas alternativas

en cuanto a la E/S y tipo de controlador respectivamente.

Hay dos formas de seleccionar opciones del menú y submenús: con teclado y con mouse. Con teclado basta con presionar la letra que corresponda a la presentada en mayúscula y subrayada. Dependiendo de la opción aparecerá un submenú o una ventana de datos. Si se trata de una ventana de datos, éstos pueden ser presentados o solicitados al usuario. Si son solicitados pueden ser valores numéricos como es el caso de los parámetros utilizados para la adquisición de datos y la operación del controlador. Estos han sido definidos por default al momento de la compilación. Sin embargo, si el usuario lo desea estos valores pueden ser cambiados. Para aceptar el dato presentado por default basta con presionar ENTER; si se desea cambiarlo se presiona BACKSPACE y se ingresa eldeseado. Este procedimiento es el mismo para todos los datos numéricos pedidos en la ventana. Al final de ella aparece subrayado el ícono OK. Si se presiona ENTER se les asignan los nuevos valores a los parámetros mostrados en pantalla. Pero si se aprieta ESC se abandonará la ventana y no se les habrá dado entrada a los datos. De hecho, cualquier tipo de ventana puede abandonarse en cualquier instante al presionar ESC.

Mediante el mouse pueden activarse opciones al posicionar el cursor en la misma y presionarse su botón izquierdo. El ícono OK de las ventanas de ingreso de datos puede también ser aceptado de la misma forma. Lo anterior implica que el botón izquierdo, utilizado para seleccionar, equivale a ENTER, mientras que el derecho, empleado para deseleccionar, equivale a ESC.

En el caso de **Controladores**, el ícono OK sirve no sólo para dar asignación a nuevos valores sino también para seleccionar el tipo de controlador. Para el controlador

Manual la entrada es directa mientras que para los demás la entrada se da mediante Correr... .

En general, la secuencia de pasos para la operación en línea se resume de la siguiente manera :

1) Posicionarse en el directorio de trabajo "CDOBJECT" y teclear CDOBJECT.
2) Seleccionar Entrada/salida [Tarjeta PCL-812] y aceptar, o ingresar, los canales A/D y D/A así como la frecuencia de muestreo.
3) Salir de la opción anterior y abrir la de Controladores [controlador deseado]. Se mostrará la ventana de ingreso de datos y se procede a dar los parámetros del controlador. Este será seleccionado al aceptar el ícono OK siendo sólo así como se podrá echar a andar la acción de control, con la excepción del de tipo manual que opera al ser elegido.
4) Concluído el paso anterior se elige la opción Correr... y se entra a la operación en línea.

Las figuras V.2, V.3 y V.4 muestran las pantallas presentadas al realizar los pasos 2,3 y 4 anteriores respectivos.

Al entrar en línea pueden observarse junto con las gráficas de control los valores tomados por parámetros de los controladores. A la vez aparece un menú cuyas opciones sólo pueden ser escogidas con mouse, sirviendo aquellas para el intercambio de graficación de las señales de control, la alteración de los parámetros del controlador y la presentación de una breve historia de control del proceso. La variable del proceso posee un rango de graficación variable dentro del intervalo (-10,10), mientras que la señal de control lo tiene dado en porcentajes (0,100%). Esto se aplica

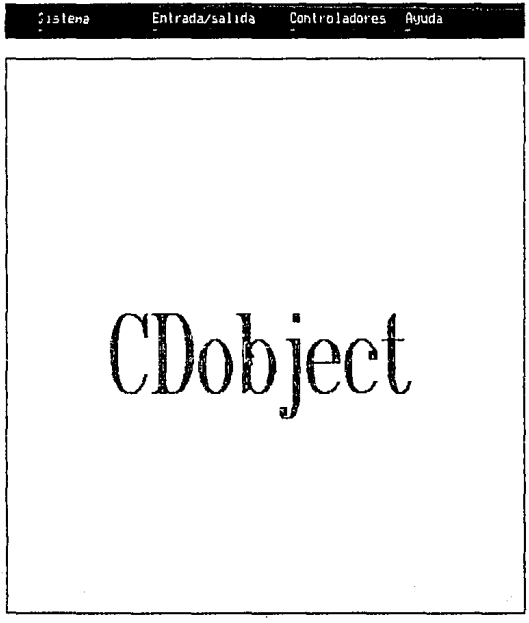


FIGURA V.1 Pantalla de entrada de CDobject

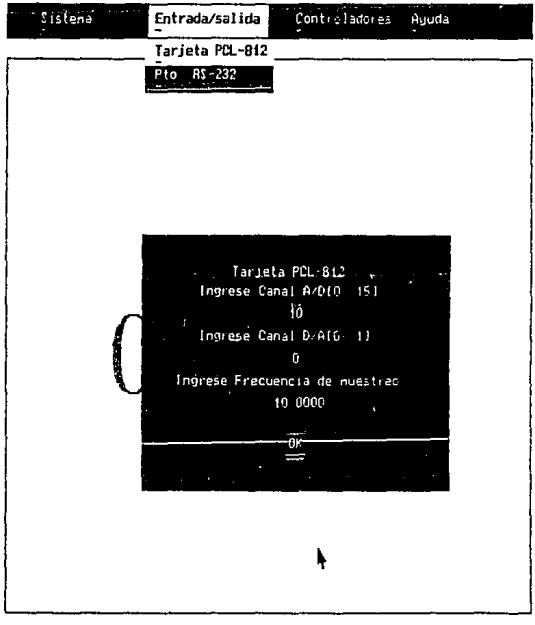


FIGURA V.2 Ventana para entrada de parámetros E/S

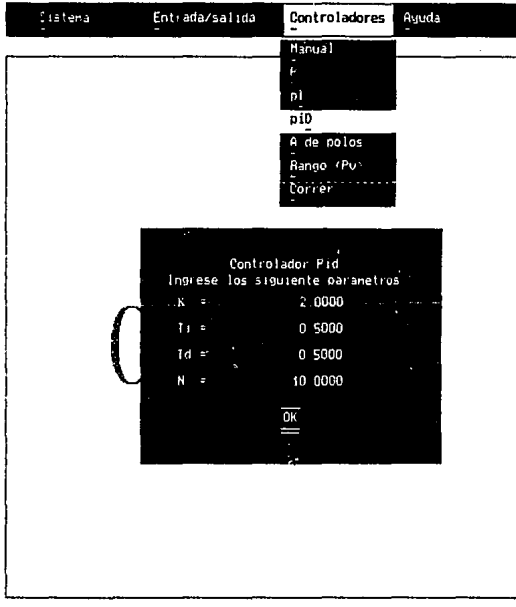


FIGURA V.3 Ventana para entrada de parámetros de controlador PID

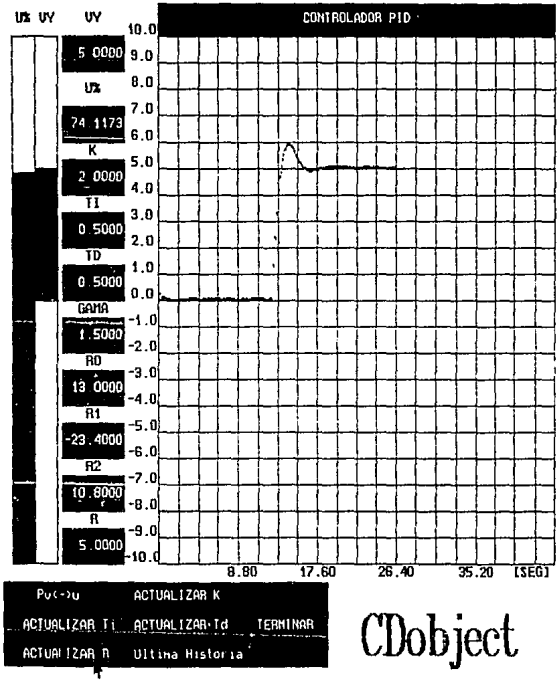


FIGURA V.4 Corrida del controlador PID

también a las gráficas de barras que en el caso de yCkT) su sentido depende de su polaridad. Para uCkT) su sentido es único.

La pantalla de graficación posee un área de 440x360 pixeles. Cuando se varía el rango de la variable yCkT), se altera la escala de ésta únicamente para su presentación. Esto quiere decir que para la conversión A/D la tarjeta PCL-812 siempre emplea 12 bits para las lecturas en el mismo rango de C-10,10) volts.

c) Modos de operación :

Bajo la presente sección se explican detalladamente todas las opciones del paquete.

* **Sistema** : Ofrece servicios de operación del disco, hora y fecha e información de CDobject evitando que se tenga que abandonar el programa para solicitarlos vía sistema operativo. Posee las siguientes sub-opciones :

** **Dir** : Visualización del presente directorio de trabajo. No muestra archivos ocultos o del sistema.

** **Cambiar dir** : Cambio del directorio de trabajo. Es necesario ingresar el trayecto completo.

** **cOpiar** : Copiado de archivos. Ambos trayectos deben ser completos.

** **Salir** : Abandonar el programa.

** **Hora y fecha** : Proporciona la hora y fecha del sistema.

** **Acerca...** : Origen del paquete.

* **Entrada/salida** : Configurar la E/S del controlador. Actualmente posee 2 opciones de las cuales sólo una es operativa :

** Tarjeta PCL-812 : Solicita los canales A/D, D/A y frecuencia de muestreo para E/S por tarjeta PCL-812. Los valores por default son : 10 para canal A/D, 0 para canal D/A y frecuencia de muestreo de 10 Hz. El rango de canales A/D es de 0 a 15, para D/A de 0 a 1 y para la frecuencia de muestreo se aceptan valores entre 0.1 y 20 Hz.

** Pto. RS-232 : No disponible .

* Controladores : Muestra los controladores disponibles y el paso a ejecución en línea. Las condiciones iniciales del sistema son nulas y los valores por default para los distintos tipos de controlador son :

K = 1	Constante de proporcionalidad
T_i = 100	Tiempo integrativo
Td = 0.1	Tiempo derivativo
N = 10	Cte. del filtro paso-bajas
Mp = 0.15	Sobrepaso de la respuesta
Tr = 1	Tiempo de elevación de resp.
A = 0.9	Primer polo de $G_p(z)$
B = -0.9	Unico cero de $G_p(z)$
C = 0.9	Segundo polo de $G_p(z)$

** Manual : La señal de referencia, que equivale a la de control, es dada manualmente por el usuario .

** P : Asignación de valor a K y selección del controlador proporcional.

** pI : Asignación de valores a K y T_i, selección del controlador proporcional-integrativo.

** piD : Asignación de valores a K, T_i, Td y N;selección del controlador porporcional-integrativo-derivativo.

** **Apolos** : Asignación de valores a M_p, T_r, A, B y C; selección del controlador por asignación de polos con cancelación de ceros para proceso de segundo orden.

** **Rango <Pv>** : Se especifica el rango de la variable del proceso, lo cual equivale también para la señal de referencia.

** **Correr...** : Ejecución en línea del controlador seleccionado. En caso de haber utilizado un controlador anterior y haber escogido otro, las condiciones del sistema se mantienen.

* **Ayuda** : Presentación en pantalla de un texto similar al presente para una mejor comprensión del paquete para su manejo. Mediante el mouse o las teclas ^ y v del cursor se puede desplazar a través de ella.

CAPITULO VI

GUIA DE MANTENIMIENTO

La presente guía tiene como propósito el explicar aspectos relacionados con la organización del programa a nivel de archivos, las opciones de compilación y ligado, y la estructura del ejecutable actual.

a) Organización del programa a nivel de archivos .

Al desarrollar software de cierto tamaño y complejidad es deseable que su estructura organizacional facilite su mantenimiento y evolución. Los programas grandes y complejos que ocupan un sólo archivo corren un mayor peligro de sufrir errores al momento de su mantenimiento, además de que el tiempo de compilación es muy grande ya que éste proceso se realiza para el programa entero.

Los compiladores actuales comerciales para lenguajes de alto nivel han tratado de remediar este aspecto. En el caso de los compiladores de C y C++ se propone una estructura de organización del programa a nivel de archivos consistente en la dispersión ordenada del código en ellos, recibiendo aquella el nombre de "proyecto".

Un proyecto es un conjunto de archivos que mantienen cierta interdependencia y que contienen la enunciación del código y el dato dados por el usuario, la definición del mismo y un programa principal. Al estar el código fragmentado en varios archivos se reducen tanto el tiempo de compilación como los errores que pudieran generarse durante ésta.

En el caso de CDobject , el proyecto está formado de la siguiente manera :

* Un archivo de encabezamiento (header) que contiene la

enunciación de las estructuras de código-dato (clases) estableciendo con claridad los elementos que las componen, sus propiedades y reglas de herencia. Nombrado CDOBJECT.HPP .

* Los archivos donde se encuentran las operaciones correspondientes a cada clase establecida en CDOBJECT.HPP son los siguientes :

TARJETA.CPP	Operaciones de la clase	Tarjeta.
COMUNICA.CPP	" " " " "	Comunica.
MANUAL.CPP	" " " " "	Manual.
P.CPP	" " " " "	P.
PI.CPP	" " " " "	Pi.
PID.CPP	" " " " "	Pid.
APOLOS.CPP	" " " " "	Apolos.
ALGO_CD.CPP	" " " " "	Algo_cd.
TECLADO.CPP	" " " " "	Teclado.
MONITOR.CPP	" " " " "	Monitor.
MOUSE.CPP	" " " " "	Mouse.
INTFACE1.CPP	" " " " "	Interface1.
INTFACE2.CPP	" " " " "	Interface2.
SISTEMA.CPP	" " " " "	Sistema.
AYUDA.CPP	" " " " "	Ayuda.

* El código fuente del programa principal se encuentra en el archivo CDOBJECT.CPP, encontrándose ahí definidas las operaciones en y fuera de línea.

Debido a que tanto la compilación como el ligado de un programa en C y C++ dependen de la inclusión de los prototipos de funciones mediante la directiva al preprocesador "# include <archivo.h>", la interdependencia del programa se asegura también al incluir en todos los archivos *.CPP ya mencionados la misma directiva pero con CDOBJECT.HPP, es decir, " # include "cdobject.hpp" ".

Anteriormente, con compiladores más antiguos, era necesario que el programador mismo creara un archivo que estableciera el proyecto (archivo.PRJ), el cual define los archivos que lo componen. En la actualidad eso ya no es así. Los compiladores actuales proveen plataformas de desarrollo bastante completas (IDE). Estas crean el archivo de proyecto en cierto formato con ciertas opciones de compilación y ligado mencionando los nombres de los archivos junto con la cantidad de código, dato y líneas encontrados durante la compilación y el ligado.

Al abrir un proyecto dentro de una IDE se ingresan sólo los nombres de los archivos con extensión CPP. Y se procede a compilarlos.

b) Opciones de compilación y ligado .

Definen cómo es generado el código máquina, en que forma es optimizado, cómo es conjuntado y qué información extra puede contener para efectos de depuración.

A continuación se listan las condiciones bajo las cuales CDobject fué compilado. Sólo se listan las opciones seleccionadas, no los menús completos del compilador :

* CODE GENERATION :
  - TEST STACK OVERFLOW
  - LARGE MODEL
  - FLOATING POINT EMULATION
  - INSTRUCTION SET 80286
  - CALLING CONVENTION C
  - GENERATE UNDERBARS
  - DEBUG INFO IN OBJ's
  - TREAT ENUMS AS INTS
  - FAST FLOATING POINT

* C++ OPTIONS :
  - SMART C++ VIRTUAL TABLES
  - CPP EXTENSION ONLY
  - OUT-OF-LINE INLINE FUNCTIONS
* OPTIMIZATIONS :
  - NONE VARIABLES REGISTER
  - OPTIMIZE FOR SPEED

* SOURCE :
  - TURBO C++ KEYWORDS
  - IDENTIFIER LENGTH = 32

* NAMES :
  - NO NAMES FOR EACH SEGMENT

* MESSAGES :
  - STOP AFTER 25 ERRORS
  - STOP AFTER 100 WARNINGS
  - DISPLAY WARNINGS

Para el ligado se dieron las siguientes opciones :
  - DETAILED MAP OF FILES
  - DEFAULT LIBRAIRIES
  - GRAPHICS LIBRAIRIES
  - "NO STACK" WARNING OFF
  - CASE SENSITIVE LINK

A pesar de que en las opciones de compilación se incluyó la de información para depuración presente en los archivos objeto, dentro de la opción DEBUGGER se establece lo siguiente para que realmente tenga efecto :

  - SOURCE DEBUGGING ON

Dentro de la gama de alternativas seleccionadas anteriores existen algunas que por la naturaleza del software

no deben ser alteradas. Tal es el caso de :

- LARGE MODEL : El modelo de compilación es LARGE debido a que el segmento de código excede los 64 Kbytes.
- GENERATE UNDERBARS : Antepone a cada identificador global (variable o función) el caracter de subrayado(_). Necesario si durante el ligado se emplean las librerías por default.
- FAST FLOATING POINT : Optimiza las operaciones de punto flotante para una mejor ejecución.
- SMART C++ VIRTUAL TABLES : El compilador crea tablas en cada archivo objeto de tal forma que al realizarse el ligado el ejecutable contiene el código más pequeño y eficiente.
- NONE VARIABLES REGISTER : No se permite el uso de variables registro debido al uso de la palabra reservada "interrupt" en MOUSE.CPP .
- OPTIMIZE FOR SPEED : Optimización del código por el compilador para una ejecución más rápida.
- TURBO C++ KEYWORDS : El compilador sólo reconoce como palabras reservadas las correspondientes a TURBO C++ .
- NO NAMES FOR EACH SEGMENT : No se le asignan nombres a los segmentos del programa.
- DEFAULT LIBRAIRIES : El compilador realiza la búsqueda del código objeto en las librerías del compilador.
- GRAPHICS LIBRAIRIES : Permite la búsqueda automática de aechivos BGI y CHR.
- "NO STACK" WARNING OFF : No se permite el despliegue del mensaje de advertencia "NO STACK" debido a que ésta opción no debe emplearse cuando se hace uso de la palabra reservada "interrupt".
- CASE SENSITIVE LINK : Permite que el compilador

distinga entre palabras mayúsculas y minúsculas, siguiendo la filosofía de C/C++ .

c) Estructura del ejecutable actual .

Puede explicarse fácilmente mediante el mapa del programa. Presenta la estructura de éste dividida en los siguientes segmentos: de código, de datos, de información para el intercambio dinámico de segmentos, de stack, etc. Todo lo anterior es mostrado en la figura VI.1 . Consta de una tabla con columnas referentes a las direcciones de inicio y paro de determinado segmento con su respectiva longitud, especificándose también la naturaleza de los segmentos.

Al observarlo puede apreciarse en primera instancia del porqué el programa fué compilado bajo el modelo LARGE. Usualmente se le asigna un segmento de 64Kbytes como máximo al código generado durante la compilación y ligado. A este segmento se le denomina _TEXT. Cuando se excede el límite ya mencionado se crean otros segmentos que forman parte del mismo segmento de código anteponiéndole el nombre del módulo o archivo al cual pertenece ese código.

De la misma forma se tiene que como otro segmento de código está el correspondiente a la emulación del coprocesador matemático 80x87. Si no se hubiera escogido la opción de emulación de punto flotante (que se puede) este segmento no aparecería en el mapa del programa.

Existe un grupo de segmentos que no corresponden ni al código, dato, stack y heap y que son utilizados en caso de programas extremadamente grandes. Los compiladores modernos para programación orientada a objetos traen consigo una innovación tecnológica : la administración dinámica de segmentos mejor conocida como tecnología VROOM (Virtual

START	STOP	LENGTH	NAME	CLASS
00000h	09627h	09628h	_TEXT	CODE
09628h	09939h	00312h	TARJETA_TEXT	CODE
0993Ah	09A0Bh	000D2h	COMUNICA_TEXT	CODE
09A0Ch	09B89h	0017Eh	MANUAL_TEXT	CODE
09B8Ah	09C2Eh	000A5h	P_TEXT	CODE
09C2Fh	09D10h	000E2h	PI_TEXT	CODE
09D11h	09FF8h	002E8h	PID_TEXT	CODE
09FF9h	0A2C4h	002CCh	APOLOS_TEXT	CODE
0A2C5h	0B14Ch	00E88h	ALGO_CD_TEXT	CODE
0B14Dh	0B8CCh	00780h	TECLADO_TEXT	CODE
0B8CDh	0E223h	02957h	MONITOR_TEXT	CODE
0E224h	0E8C2h	0069Fh	MOUSE_TEXT	CODE
0E8C3h	123E3h	03B21h	INTFACE1_TEXT	CODE
123E4h	12EF4h	00B11h	SISTEMA_TEXT	CODE
12EF5h	130D7h	001E3h	AYUDA_TEXT	CODE
130D8h	14E92h	01DBBh	INTFACE2_TEXT	CODE
14E93h	197A3h	04911h	CDOBJECT_TEXT	CODE
197B0h	1BF16h	02767h	EMU_PROG	CODE
1BF20h	1C4E7h	005C8h	E87_PROG	CODE
1C4F0h	1C4F0h	00000h	_FARDATA	FAR_DATA
1C4F0h	1C4F0h	00000h	_OVERLAY_	OVRINFO
1C4F0h	1C507h	00018h	_INIT_	INITDATA
1C508h	1C508h	00000h	_INITEND_	INITDATA
1C508h	1C50Dh	00006h	_EXIT_	EXITDATA
1C50Eh	1C50Eh	00000h	_EXITEND_	EXITDATA
1C510h	1C510h	00000h	_1STUB_	STUBSEG
1C510h	1E3E3h	01ED4h	_DATA	DATA
1E3E4h	1E3E7h	00004h	_CVTSEG	DATA
1E3E8h	1E3F3h	0000Ch	_SCNSEG	DATA
1E3F4h	1E3F4h	00000h	GRAPH	DATA
1E3F4h	1E4C7h	000D4h	_BSS	BSS
1E4C8h	1E4C8h	00000h	_BSSEND	ENDBSS
1E4D0h	1E65Fh	00190h	_STACK	STACK

FIGURA VI.1 MAPA DE SEGMENTOS DEL PROGRAMA CDobject .

Run-Time Object-Oriented Memory Manager). Consiste en lo siguiente : cuando la memoria RAM no alcanza para ejecutar cierto proceso (y hablando tan sólo de un ambiente monousuario como DOS), es necesario ejecutar un intercambio (swappeo) dinámico de segmentos entre el almacenamiento secundario y el principal. Para poder ejecutar éste almacenamiento virtual es necesario que el administrador de "overlays" sea lo suficientemente inteligente para que permita el llamado de funciones pertenecientes a otro segmento o unidad-overlay que no se encuentra cargado en ese momento. Para ésto el administrador busca espacio para cargar esa unidad reubicando si es necesario las otras ya asignadas en RAM. Este esquema es el comúnmente utilizado por ambientes como WINDOWS para la ejecución de sí mismo y sus aplicaciones. En el caso del presente paquete no fue necesario utilizar esta técnica pero no está de más mencionarla para efectos de futuros desarrollos.

Aparecen también los segmentos correspondientes a los datos y al stack(pila). En el caso de los datos, éstos se dividen en los segmentos _DATA(datos globales inicializados) y _BSS(datos globales no inicializados). El stack ocupa un sólo segmento.

La figura VI.1 representa el inicio de un archivo con información más detallada generada durante la compilación y que permanece bajo el nombre "CDOBJECT.MAP" en el subdirectorio de trabajo del paquete. Proporciona la ubicación que tiene cada clase dentro de los segmentos así como las operaciones inherentes a ellas. Todos estos datos pueden ser utilizados por un depurador.

CAPITULO VII

CONCLUSIONES

Las siguientes conclusiones tratan acerca de las experiencias aprendidas durante el desarrollo del presente trabajo, los alcances de éste y la tendencia a futuro que presenta la programación orientada a objetos (POO) .

* Al momento del análisis del software, y su programación, surgieron inevitables las comparaciones entre la programación estructurada usual y la orientada a objetos. Esta última, merced a sus tres propiedades principales(objetos, polimorfismo y herencia), representa, forzando los términos, una forma bastante determinística de programación. Es cierto que sobre el programador, y sobre todo en un lenguaje tan flexible como C++, recaen sobremanera los resultados obtenidos. Sin embargo, la POO facilita el desarrollo del programa y ulteriores modificaciones al obligar al desarrollador a plantear en una forma concienzuda la naturaleza y características de aquél. Así, al reconocer y diferenciar las entidades código-dato correctamente y asociarles un nombre particular, es posible afirmar que el grado de estructuración es alto, lo cual redunda en una disminución del tiempo de desarrollo así como una gran capacidad de mantenimiento y evolución. La herencia de las entidades (clases) y la posibilidad de aplicar el polimorfismo en las operaciones facilita la comunicación entre módulos y el manejo de operaciones también respectivamente.

* Por mantenimiento se entiende como aquellos cambios requeridos por el usuario que son originados por las necesidades de éste que van surgiendo al pasar el tiempo. Por evolución se entiende al desarrollo de nuevos programas en base a elementos del código fuente que no

son específicos a la naturaleza de un sólo programa sino que presentan un intervalo muy amplio de utilización.

* Es así como la POO logra que el ciclo Reducir-Reusar-Reciclar código llegue a darse en forma efectiva. La reducción se da al momento de la programación, la reutilización con el mantenimiento y el reciclado al darse la evolución.

* Anteriormente se señaló que el presente paquete posee fines puramente académicos. Esto es correcto si se trata sólo de su utilización por el usuario. Sin embargo, desde el punto de vista del programador sus alcances son más amplios. Pueden conocerse a través de su exámen la filosofía y fundamentos de POO. Por otra parte, debido a la variedad de aspectos que abarca, como operaciones de disco, información del sistema, uso extensivo de dispositivos de E/S (monitor,teclado,mouse), adquisición de datos mediante conversiones A/D y D/A, algoritmos de control digital, etc, es relativamente fácil emplear codigo del programa para otras aplicaciones que, aunque no tengan nada que ver con aspectos de control digital, no obsta en forma alguna.

* A lo largo de la programación se trató siempre de lograr un equilibrio entre la eficacia y la eficiencia, es decir, entre lo bueno y lo mejor. Dentro del medio de la programación se afirma con insistencia que lo eficaz(bueno) es enemigo de lo eficiente(mejor) y viceversa. Por tal motivo lo eficaz se buscó a través de la adecuada elección de estructuras de decisión y control, mientras que lo eficiente está representado por la serie de restricciones y/o protecciones del código para condiciones críticas así como la estructura de clases establecida.

* Continuamente se ofrecen en el mercado versiones más

avanzadas de compiladores para C++. Estos ofrecen nuevas características, como el polimorfismo a nivel de clases, que permiten incrementar la potencia del código, reducir su tamaño y tiempo de ejecución.

* En el presente es de resaltar que un gran número de aplicaciones han sido creadas para su ejecución bajo ambientes más amigables que DOS como lo es WINDOWS. Para ello los nuevos compiladores proveen una biblioteca completa de funciones que permiten la programación en WINDOWS. Es así como pueden crearse programas con interfaces muy amigables sin demasiado esfuerzo por parte del programador, lo cual es importante ya que la mayor parte del presente software se dedica a la función de interface.

* A un plazo cercano se tiene estimado que saldrán al mercado sistemas operativos, para equipos personales y multiusuarios, enfocados a la programación orientada a objetos. Esto quiere decir que los programas que componen el sistema operativo (administradores de memoria, de procesador, de dispositivos E/S, etc) habrán sido creados con el propósito principal de ejecutar software POO. Por lo tanto la POO representa el futuro forzoso al cual deriva desde este momento la programación, no importando el tipo de aplicación a desarrollar.

BIBLIOGRAFIA

* Schildt, Herb .
" TURBO C/C++ : the complete reference " .
Ed. Osborne-Mcgraw-Hill . Second Edition . 1992 .

* Atkinson & Atkinson, Lee and Mark .
" USING BORLAND C++ " .
Ed. QUE, Programming Series . 1991 .

* Astrom Karl Johan, Wittenmark Björn .
" COMPUTER CONTROLLED SYSTEMS : THEORY AND DESIGN " .
Ed. Prentice-Hall, Englewood Cliffs N.J. . 1984 .

* Advantech Co., Ltd .
" PC-LabCard 812 USER'S MANUAL " .
Advantech Co., Ltd . Taiwan, Dec. 1989 .

* Profesores del Departamento de Ingeniería de Control .
Facultad de Ingeniería . U.N.A.M.
" MANUAL DEL LABORATORIO DE CONTROL DIGITAL " .
Semestre 92-1 . Facultad de Ingeniería . U.N.A.M.

INDICE DE FIGURAS

INDICE DE DIAGRAMAS DE FLUJO