86

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESJ

FACULTAD DE INGENIERIA

CONTROLADOR DIGITAL BASADO EN UN MICROCONTROLADOR MC68HC11

T E S I S

Que para obtener el Título de:

INGENIERO EN COMPUTACION

P r e s e n t a n:

OMAR FELIPE QUIROZ MORENO

OSCAR ABELARDO RODRIGUEZ RESENDIZ

DIRECTOR DE TESIS

ING. RICARDO GARIBAY JIMENEZ

México, D.F. 1995

INDICE

I. INTRODUCCIÓN

I.-INTRODUCCIÓN

Aunque el principio del control es usado en una amplia variedad de procesos, no es una invención técnica. Es un fenómeno natural que permite que en un estado particular sea retenido automáticamente a pesar de las interferencias externas. Muchos procesos biológicos y ambientales, así como también económicos e incluso sociales funcionan acorde a este principio.

A lo largo del tiempo la ingeniería de control se ha desarrollado como una disciplina autónoma aplicable a diversos campos.

Podemos decir que un sistema de control es un grupo de componentes acoplados de cierta manera que permitan regular una energía de entrada para alcanzar una respuesta deseada. El sistema puede ser de tipo eléctrico, neumático, mecánico, hidráulico, etc., o una combinación de ellos, la energía de entrada se refiere a una variable física tales como temperatura, voltaje, presión, flujo.

Un sistema de control ideal es aquel donde la salida es función directa de la entrada, sin embargo, en la práctica existen perturbaciones que afectan a la salida. La naturaleza de estos disturbios cambia de sistema en sistema, pero es de gran importancia identificarlos porque una vez que se conozcan, es posible modificar el sistema para minimizar sus efectos y, en algunos casos, hasta eliminarlos.

La salida de control depende de que también esté sintonizada la respuesta del controlador a las características del proceso a controlar. Esta sintonización del controlador de acuerdo a los requerimientos es el principal problema de la ingeniería de control.

Diferentes descripciones matemáticas en el dominio del tiempo y la frecuencia han sido usadas para desarrollar procedimientos teóricos de control. La representación de la función de transferencia es usada comúnmente para describir el comportamiento de una planta o proceso.

Una de las configuraciones mas usadas para llevar a cabo control es la que se conoce como sistema de lazo cerrado, donde la salida del sistema físico a controlar o planta es llevada nuevamente hacia a la entrada, con el fin de compararla con ella y obtener una diferencia o señal de error, dicha señal de error es procesada mediante un dispositivo llamado controlador, el cual

produce una entrada a la planta que intenta reducir a cero la señal de error, es decir, trata de llevar la salida al valor deseado.

Los primeros sistemas de control eran en su mayoría sistemas de tipo analógico. Dichos sistemas son relativamente complejos e incómodos de diseñar y mantener. Sin embargo con el desarrollo de la tecnología digital y la invención de los circuitos integrados, el diseño de los sistemas de control se hizo mas fácil y más económico. Actualmente, la mayoría de los sistemas de control se basan en microprocesadores, principalmente por la disponibilidad de circuitos integrados (microcontroladores), memorias baratas y la gran capacidad en el manejo de datos con que cuentan.

La principal característica de un controlador digital es la implementación de la función de control mediante un algoritmo en forma de programa. Comparado con la operación continua de un controlador analógico el cual actúa prácticamente sin retardo, un controlador digital puede solamente procesar el valor actual y generar un nueva señal en puntos discretos en el tiempo. Convertidores analógicos digitales y digitales analógicos son necesarios para permitir la conexión con la planta o proceso.

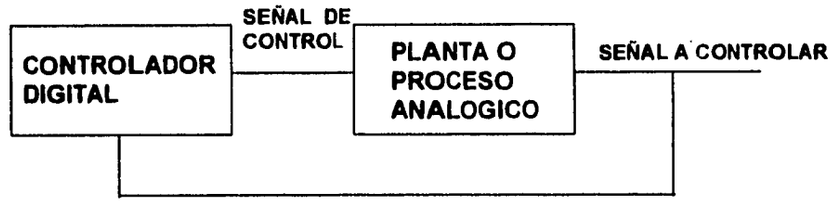


figura 1.1

En el trabajo presentado se elaboró un dispositivo digital que realiza las funciones de control de lazo cerrado, para manejar una malla de control a través de los convertidores propios del microcontrolador usado que es el MC68HC11F1. Además este dispositivo dispone de una pantalla pequeña para realizar la configuración, y operación del controlador en forma local, así como presentar de manera gráfica la respuesta del sistema controlado, el diagrama de bloques del controlador digital se presenta en la figura 1.2.

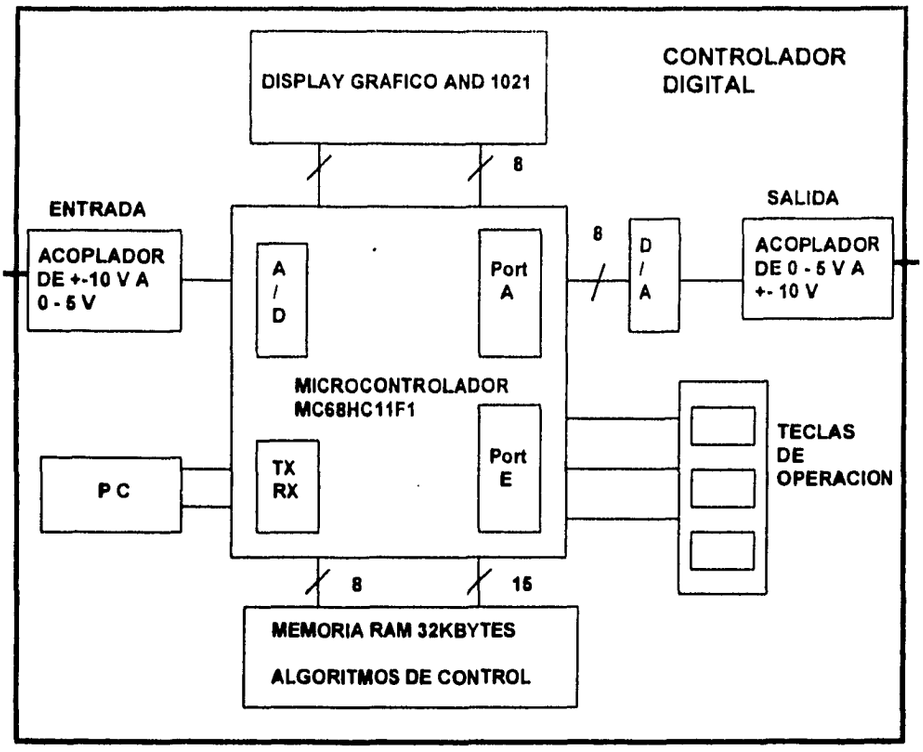


figura 1.2 Diagrama de bloques del controlador digital

En relación con los algoritmos de control se incorporó además del clásico PID (Proporcional Integral Derivativo) una variante más avanzada como lo es el algoritmo de asignación de polos y ceros. Los rangos de entrada y salida son de +- 10V.

El controlador digital está diseñado de tal manera, que además de llevar a cabo un control real, es un instrumento didáctico y es de gran utilidad para comprender el funcionamiento de los diferentes algoritmos de control, ya que debido a su programación modular es fácil poder programar otros algoritmos además de los ya implementados.

Este diseño puede servir para originar nuevas arquitecturas más complejas gracias a los componentes utilizados como lo es el microcontrolador que tiene bastantes funciones que no se utilizaron en este diseño, y con el continuo desarrollo de otros microcontroladores y procesadores de señales se podrán implementar sistemas digitales más complejos.

En este trabajo se describe el diseño e implementación del sistema digital en cuestión de la siguiente manera, en la capítulo 2 explica toda la teoría y cálculos realizados previamente, para la obtención de los algoritmos de control utilizados que son el Integral-Proporcional-Derivativo y el de asignación de polos y ceros, así como una simulación de este último.

En el capítulo 3 se explica algunas de las características principales del microcontrolador MC68HC11 que fueron utilizadas así como otras que se pueden utilizar.

En el capítulo 4 se describe detalladamente todo el diseño e implementación de todo el sistema digital, se habla de los periféricos utilizados, como lo son acopladores de señales, el convertidor digital analógico, la pantalla de despliegue, las teclas para la operación, la memoria externa, así como la programación de estos y de los algoritmos de control.

Finalmente en el capítulo 5 se realizan pruebas para verificar el funcionamiento del dispositivo con ayuda de simuladores para computadora, así como un simulador de procesos analógicos del Laboratorio de Control Digital de la Facultad de Ingeniería UNAM.

En el diagrama de bloques del dispositivo diseñado (figura 1.2), se puede apreciar de una manera global los periféricos utilizados.

II.-ALGORITMOS DE CONTROL DE LAZO CERRADO

II.-ALGORITMOS DE CONTROL DE LAZO CERRADO

Dentro de los algoritmos de control de lazo cerrado podemos encontrar entre otros los siguientes:

Algoritmos PID.

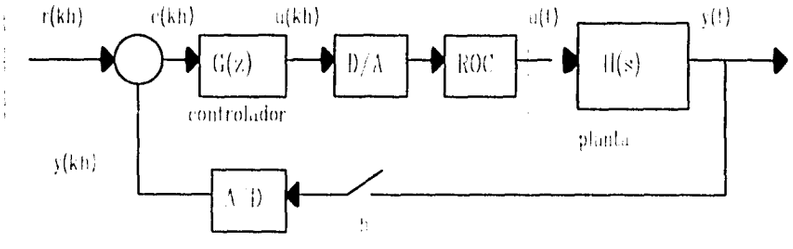
Asignación de Polos y Ceros.

II.1 Algoritmos PID.

La teoría de control considera tres acciones básicas de control: la acción proporcional, la acción integral y la acción derivativa. Estas acciones de control se pueden aplicar de manera combinada dando lugar a diferentes tipos de controladores.

El controlador proporcional (P), el proporcional-integral (PI) y el proporcional-integral-derivativo (PID), son tres de los algoritmos más populares de control. Su importancia es mayor, no sólo por el hecho de que son tradicionalmente estudiados, sino también porque dan solución satisfactoria a múltiples aplicaciones en la industria.

FIGURA 2.1



En la figura 2.1 se muestra un sistema típico de control digital.

Donde :

H(s) es la función de transferencia continua de la planta.

G(z) es la función de transferencia discreta del controlador.

r(kh) es la secuencia de referencia.

e(kh) es la secuencia de error.

u(kh) es la secuencia de control.

y(t) es la salida de la planta.

h es el período de muestreo.

k es elemento de los enteros no negativos.

El algoritmo de control que conjunta las tres acciones de control mencionadas, es el que corresponde al controlador proporcional-integral-derivativo. Este controlador es el de empleo más frecuente en diversas aplicaciones debido a que utiliza acciones de control complementarias entre sí, por lo que mejora precisión y estabilidad del sistema en malla cerrada.

La siguiente expresión describe al controlador PID de estructura estándar (el controlador PID tradicional).

$$G'(s) = \frac{U(s)}{E(s)} = K\left(1 + \frac{1}{Ti\,s} + \frac{Td\,s}{1 + (Td/N)s}\right)$$

donde: Ti es el tiempo de integración

Td es el tiempo de derivación

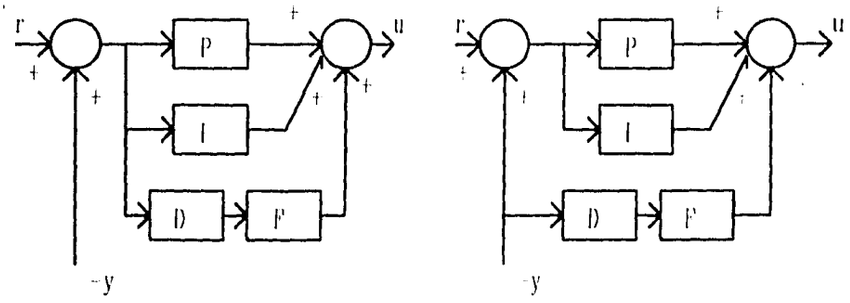
K es la ganancia.

La aproximación discreta de la acción PID se efectúa empleando el método de 'diferenciación hacia adelante' para la acción integrativa y la 'diferenciación hacia atrás' para la acción derivativa y el filtro.

La ecuación de diferencias total que define el algoritmo de control se obtiene adicionando las componentes obtenidas en la aproximación discreta; sin embargo, en la práctica, es necesario realizar el agrupamiento de acuerdo con algunas estructuras algebraicas particulares que permitan

formas de operación confiables. Las estructuras mas frecuentemente empleadas se muestran el la figura 2.2 en forma de diagramas de bloques.

FIGURA 2.2



ESTRUCTURA (a) ESTRUCTURA (b)

La estructura (a) corresponde a la simple adición de los algoritmos elementales; sin embargo, presenta la gran desventaja de que la operación derivativa se aplica sobre el error. Esto es inconveniente, debido a que eventuales cambios intempestivas de esta variable, ocasionan la presencia de 'picos' en la señal de control. Esta situación es muy factible si se piensa que los cambios de referencia se introducen frecuentemente como señales de tipo escalón.

La estructura (b) es mejor debido a que la derivativa se aplica sobre la variable de proceso directamente, la cual es una variable lenta normalmente, evitando así, posibles picos en la señal de control. También se dice que con esta estructura el efecto 'anticipatorio' se aplica efectivamente sobre la variable que lo requiere, es decir, las variaciones del proceso.

II.1.1 Obtención del algoritmo o ecuación en diferencias

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

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

con:

$$\alpha = \frac{K\,h}{Ti} \qquad \frac{1}{\beta} = \frac{K\,Td}{h} \qquad \gamma = \frac{Td}{N\,h} + 1$$

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

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

Para la estructura (a) resulta : T(z) = Q(z), donde los coeficientes de este polinomio son:

$$q_0 = K\gamma + 1/\beta$$

$$q_1 = k(1 - 2\gamma) + \alpha\gamma - 2/\beta$$

$$q_2 = (k - \alpha)(\gamma - 1) + 1/\beta$$

Para la estructura (b) el polinomio Q(z) es igual al definido para la estructura (a) anterior; sin embargo el polinomio T(z) se define:

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

$$t_0 = K\gamma$$

$$t_1 = k(1 - 2\gamma) + \alpha\gamma$$

$$t_2 = (k - \alpha)(\gamma - 1)$$

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

Estructura (a)

$$s_0 u(kh) = - s_1 u(kh - h) - s_2 u(kh - 2h) + q_0[r(kh) - y(kh)] + q_1[r(kh - h) - y(kh - h)] + q_2[r(kh - 2h) - y(kh - 2h)]$$

Estructura (b)

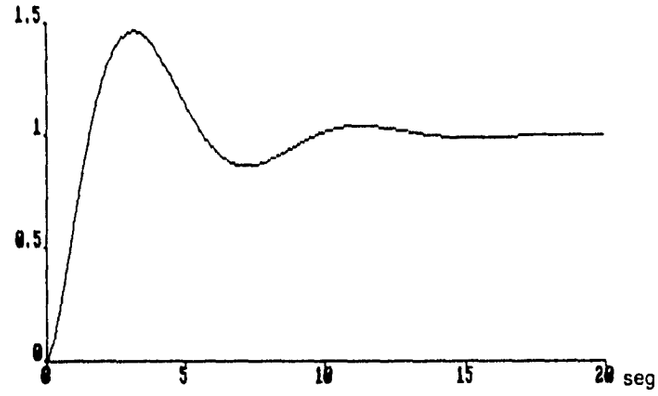
$$s_0 u(kh) = - s_1 u(kh - h) - s_2 u(kh - 2h) + t_0 r(kh) + t_1 r(kh - h) + t_2 r(kh - 2h) - q_0 y(kh) - q_1 y(kh - h) - q_2 y(kh - 2h)$$

Tomando como ejemplo la siguiente función de transferencia:

$$Gp(z) = \frac{0.001681(z + 0.96078)}{(z - 0.95122) (z - 0.93239)}$$

aplicando el algoritmo de control obtenido con un tiempo de integración Ti = 0.4s, un tiempo de derivación Td = 0.4s y una ganancia K = 0.4 se obtiene el siguiente comportamiento para la señal de control y la señal de salida:

Señal de salida:(Volts)



Señal de control:(Volts)

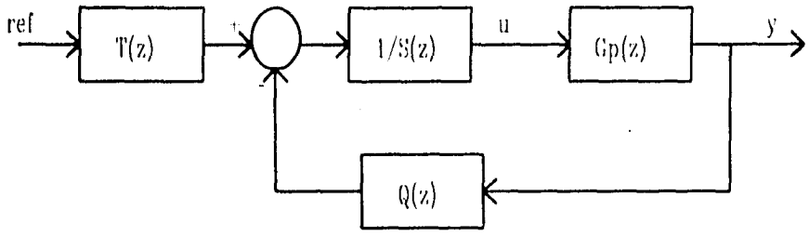
De acuerdo a las gráficas mostradas podemos concluir que es factible implementar este algoritmo ya que la señal de control generada no es muy grande en relación a la referencia.

II.2 Algoritmo de asignación de polos y ceros

El objetivo de este algoritmo es poder asignar los polos al sistema en una posición especificada por el modelo, de tal manera que obtengamos la respuesta deseada en comportamiento dinámico y de estado estable.

A continuación se mostrará cómo se obtuvo el algoritmo para una planta de segundo orden utilizado en nuestro sistema.

Considerando los diagramas de bloques de un sistema con control retroalimentando como se muestra en la figura 2.3.



donde T(z), S(z), Q(z) son polinomios en z

Tenemos que la función de transferencia de una planta conocida es :

$$Gp(z) = \frac{B(z)}{A(z)} \quad \text{..... (a)}$$

Se propone un modelo del sistema completo que satisface especificaciones del comportamiento dinámico y de estado estable:

$$y/ref = \frac{Bm(z)}{Am(z)} \quad \text{.....(b) Modelo requerido.}$$

Una vez propuesto y realizado el diseño debe observarse como:

$$y/ref = \frac{T(z)/S(z)*Gp(z)}{1+Q(z)/S(z)*Gp(z)} = \frac{Bm(z)}{Am(z)} \quad \text{..... (c)}$$

sustituyendo (a) en (b) tenemos:

$$y/ref = \frac{T(z)*B(z)}{S(z)*A(z)+Q(z)*B(z)} = \frac{Bm}{Am} \quad \ldots\ldots\ldots\ldots\ldots(1)$$

Función de transferencia de Malla Cerrada

La señal de control estará dada por la siguiente ecuación:

$$U(z) = \frac{T(z)}{S(z)} ref - \frac{Q(z)}{S(z)} y$$

El lazo de control anterior representa la combinación del control anticipatorio, del cual su función de transferencia es:

$$H(z)ff = \frac{T(z)}{S(z)}$$

y una retroalimentación con función de transferencia:

$$H(z)fb = \frac{Q(z)}{S(z)}$$

Para asegurar que la retroalimentación y el control anticipatorio son funciones de transferencias causales debe cumplirse que:

$$grad\ S >= grad\ T \qquad y$$

$$grad\ S >= grad\ Q$$

Una vez conocidos T, S, y Q se derivan las ecuaciones diferenciales de la señal de control. Esta ecuación es la que se programa en el dispositivo que realizará el control de la planta.

En el procedimiento se podrán asignar los polos en una posición especificada por el modelo. Todo el algoritmo está encaminado a ese objetivo.

Los polos en lazo cerrado de la ecuación 1 son la solución de la ecuación característica:

$$AS + BQ = 0$$

Los ceros del lazo cerrado del sistema son los ceros de los polinomios B y T.

El orden del sistema de lazo cerrado es usualmente más grande que el orden del modelo.

Para satisfacer la condición en 1 entonces debe haber cancelación de polos y ceros.

Primero consideremos los ceros en lazo abierto. Los ceros del polinomio B. Si un factor de B no es un factor de Bm entonces este debe ser cancelado por un polo en lazo cerrado, que es un factor de AS + BQ. El sistema en lazo cerrado debe ser estable, por lo tanto, los ceros estables pueden ser cancelados, ya que los ceros inestables no pueden ser eliminados. Por lo tanto, el factor B esta dado como:

$$B = B^{+} B^{-}$$

donde B^{-} son los ceros inestables, que se encuentran fuera del círculo unitario, y B^{+} son los ceros estables.

Esto trae una factorización única, el coeficiente de la potencia más alta debe ser unitario, es decir mónico.

Entonces, B^{-} no puede ser un factor de AS + BQ, por lo tanto

$$Bm = B^{-} Bm'$$

Esto implica que ceros inestables en el proceso no pueden ser cambiados, pero deben ser incluidos en Bm. Entonces B^{+} es un factor de AS + BQ, entonces B^{+} también debe ser un factor de S, por lo tanto

$$S = B^{+} S'$$

La ecuación 1 debe puede ser escrita

$$\frac{B^{+} B^{-} T}{B^{+}(A S' + B^{-} Q)} = \frac{B^{-} Bm'}{Am} \qquad (2)$$

Cancelando factores en común

$$\frac{T}{A S' + B^{-} Q} = \frac{Bm'}{Am} \qquad (3)$$

y esto hace que Am sea un factor de $A S' + B^- Q$. Además, esto hace que el polinomio observador sea cancelado en la función de transferencia de la señal de entrada a la salida. Por lo tanto, hace que Ao sea un factor de $A S' + B Q$.

Por lo tanto, se obtienen las siguientes condiciones:

$$A S' + B^- Q = Ao\ Am \quad y \quad T = Bm'Ao$$

por lo tanto

$$A S + B Q = B^+ Ao\ Am$$

De este modo los polos en lazo cerrado son los cancelados por los ceros en estado estable, B^+, el modelo de polos, Am, y los polos observadores, Ao.

Enunciado Algebraico.

El problema matemático básico es determinar que los polinomios X y Y satisfacen la ecuación lineal

$$AX + BY = C$$

donde A, B y C son polinomios conocidos

Las soluciones únicas de la ecuación se dan cuando

$$\text{grad } X < \text{grad } B \text{ ó } \text{grad } Y < \text{grad } A$$

Condiciones de causalidad

Existe una solución causal, en el método de asignación de polos y ceros si

$$\textbf{grad Am - grad Bm >= grad A - grad B} \qquad \textbf{(4)}$$

y

$$\textbf{grad Ao >= 2 grad A - grad Am - grad } B^+ \textbf{ - 1} \qquad \textbf{(5)}$$

Prueba:

Como anteriormente se había dicho

$$S = B^+ S' \text{ y } A S' + B^- Q = Ao\ Am$$

lo que nos da

$$A S + B Q = B^+ Ao\ Am$$

De acuerdo a las condiciones de causalidad de los lazos de control admisible tenemos que

grad Q <= grad S y grad B <= grad A

esto da que

grad A S= grad (AS + BQ) = grad B^+ Ao Am

ya que al multiplicar los polinomios A y S nos darán el termino de mayor grado en la ecuación.

Como:

grad A + grad S = grad (A S + B Q) = grad B^+ Ao Am

grad A + grad S = grad B^+ Ao Am

por lo tanto

grad S = grad Ao + grad Am + grad B^+ - grad A **(6)**

como habíamos visto anteriormente

T = Bm' Ao

entonces tenemos que

grad T = grad Ao + grad Bm'

y considerando las condiciones de causalidad de los lazos de control admisibles

grad S >= grad T

tenemos que

grad Ao + grad Am + grad B^+ - grad A >= grad Ao + grad Bm'

por lo tanto

grad Am - grad Bm' >= grad A - grad B^+

restando grad B^- en ambos lados nos da la primera condición de causalidad

grad Am - grad Bm >= grad A - grad B

De acuerdo al enunciado matemático tenemos que para

A S' + B' Q = Ao Am

grad Q < grad A

Escogiendo

grad Q = grad A - 1 **(7)**

y la inecuación

grad S >= grad Q

grad Ao + grad Am + grad B^+ - grad A >= grad A - 1

lo que nos resulta la segunda condición de causalidad

grad A >= 2 grad A - grad Am - grad B^+ - 1

Síntesis de solución

A) Datos de la planta polinomios A y B deben ser coprimos.

B) Especificaciones Am y Bm deben cumplir con las condiciones de causalidad, B^- debe ser factor de Bm

C) Condiciones de compatibilidad ecuación (4) y (5)

Nota: Si estas condiciones no se cumplen para cierto problema no existe solución, y es necesario reformular el modelo.

D) Cálculos

1) Factorizar el polinomio B de acuerdo a la ecuación $B = B^+ B^-$ en esta factorización se deja al diseñador ya que estos términos serán cancelados con el bloque [1/S]

2) Calcular grado de Q de la ecuación (7) y gradS de la ecuación (6)

Calcular grado de S' = gradA_0 + gradAm - gradA

resolver la ecuación A S' + B^- Q = Ao Am cuya solución será S' y Q dos polinomios que se conoce su grado., de A_0 se conoce su grado y debe proponerse que sea un polinomio mónico y estable.

E) Integración de resultados

1) Calcular S con la ecuación $S = B^+ S'$

2) Calcular T con la ecuación T = Bm'Ao

El polinomio Q es resultado directo del paso anterior los polinomios enunciados son la solución del diseño.

F) Comprobación. Si los polinomios T, S, y Q se introducen en el diagrama general del sistema y se sintetiza y/r, el resultado debe ser el modelo.

II.2.1 Obtención del algoritmo de asignación de polos y ceros, con cancelación de un cero

Planta de 2º orden $Gp(z) = \dfrac{k(z - b)}{(z - a)(z - c)}$ $\quad [a],[b],[c] < 1$

Modelo $\dfrac{Bm}{Am} = \dfrac{(1 + P_1 + P_2)\, z}{z^2 + P_1 z + P_2}$ cancelación del cero z=b

Donde:

$P_1 = -2\, e^{-\xi \omega T} \cos(\omega T (1 - \xi^2)^{1/2})$

$P2 = e^{-\xi \omega T}$

ξ = Coeficiente real de amortiguamiento.

ω = Velocidad angular del sistema

Desarrollo:

$B = B^- B^+$

$B = K (z - b)$

$B^+ = (z - b)$

$B^- = K$

Condiciones de compatibilidad :

* grad Am - grad Bm >= grad A - grad B

  2 - 1 >= 2 - 1

* grad A_0 = 2grad A - grad Am - grad B^+ - 1

  grad A_0 = 4 - 2 - 1 - 1 por lo tanto grad A_0 = 0

$A(z) = 1$

Solución

* grad Q = grad A - 1 = 1 por lo tanto $Q(z) = q0z + q1$

* grad S = grad A - 1 = 1 por lo tanto

  grad S = grad Ao + grad Am + grad B^+ - grad A

* grad S_1 = grad A_0 + grad Am - grad A

$S=B^{+}S_1$ grad $S_1 = 0$

$S_1(z) = \alpha$

$^{*}\ S_1A + QB^{-} = A_0Am$

$\alpha(z - a)(z - c) + K(q_0z + q_1) = 1(z^2 + P_1z + P_2)$

Desarrollando el lado izquierdo:

$\alpha z^2 - \alpha(a + c)z + \alpha ac + Kq_0z + Kq_1 = z^2 + P_1z + P_2$

por lo tanto :

$\alpha = 1$

$-\alpha(a + c) + Kq_0 = P_1$ por lo tanto $\mathbf{q_0 = \dfrac{P_1 + (a + c)}{K}}$

$\alpha ac + Kq1 = P2$ por lo tanto $\mathbf{q1 = \dfrac{P2 - ac}{K}}$

$Q(z) = q_0z + q_1$

$$T(z) = A_0Bm' = A_0\frac{Bm}{B^{-}} = \frac{(1 + P_1 + P_2)\,z}{K}$$

$S(z) = S_1(z)B^{+} = \alpha B^{+} = (z - b)$

Considerando la señal de control :

$$U(z) = \frac{T(z)}{S(z)}\,ref - \frac{Q(z)}{S(z)}\,y$$

Tenemos que :

$$U(z) = \frac{T_0z\ ref - q_0zy - q_1y}{s_0z - s_1}$$

$$U(z) = \frac{z(T_0\ ref - q_0y) - q_1y}{s_0z - s_1}$$

$$U(z) = \frac{(T_0\ ref - q_0y) - q_1yz^{-1}}{s_0 - s_1z^{-1}}$$

De donde se obtiene :

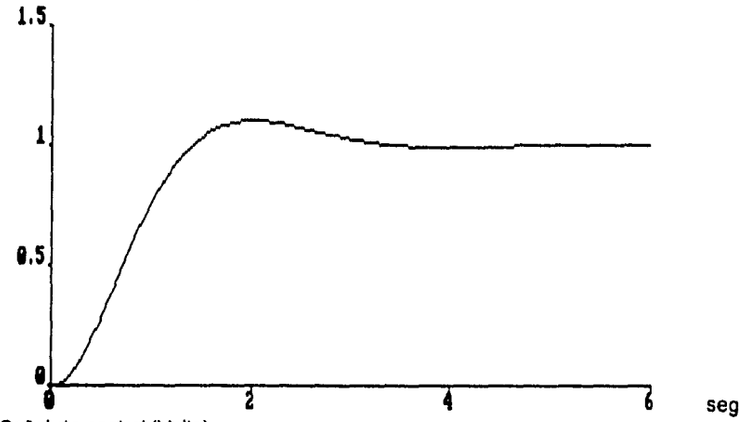
$$U[kh] = t_0R[kh] - q_0y[kh] - q_1y[kh - h] + bU[kh - h]$$

Tomando como ejemplo la siguiente función de transferencia:

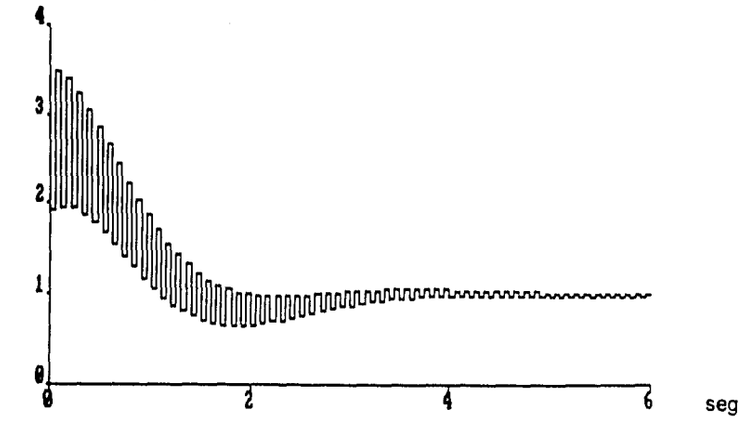
$$Gp(z) = \frac{0.001681(z + 0.96078)}{(z - 0.95122) (z - 0.93239)}$$

aplicando el algoritmo de control obtenido con un sobrepaso de Mp = 0.1 y un tiempo de levantamiento Tr = 1.4s se obtiene el siguiente comportamiento para la señal de control y la señal de salida: (con una frecuencia de muestreo de 20 Hz)

Señal de salida:(Volts)



Señal de control:(Volts)



Como se puede observar la señal de control tiene mucha oscilación en estado transitorio y valores mucho mayores a la referencia.

II.2.2 Obtención del algoritmo de asignación de polos y ceros, sin cancelación de ceros

Planta de 2º orden $Gp(z) = \dfrac{k(z-b)}{(z-a)(z-c)}$ $[a],[b],[c] < 1$

Modelo $\dfrac{Bm}{Am} = \dfrac{1+P_1+P_2}{1-b}\dfrac{(1+P_1+P_2)z}{z^2+P_1z+P_2}$ sin cancelación del cero

Desarrollo:

$B = B^-B^+$

$B^+ = 1$

$B^- = K(z-b)$

Condiciones de compatibilidad :

* grad Am - grad Bm >= grad A - grad B

  2 - 1 >= 2 - 1

* grad A_0 = 2grad A - grad Am - grad B^+ - 1

  grad A_0 = 4 - 2 - 0 - 1 por lo tanto grad A_0 = 1

$A(z) = z$

Solución

* grad Q = grad A - 1 = 1 por lo tanto $Q(z) = q0z + q1$

* grad S = grad A - 1 = 1 por lo tanto

  grad S = grad Ao + grad Am + grad B^+ - grad A

* grad S_1 = grad A_0 + grad Am - grad A

$S=B^+S_1$ grad $S_1 = 1$

$S_1(z) = z - s1$

* $S_1A + QB^- = A_0Am$

$(z-s1)(z-a)(z-c) + K(z-b)(q_0z + q_1) = z^3 + P_1z^2 + P_2z$

Poniendo z = b determinamos s1:

$$s1 = -b + \frac{b(b^2 + P_1 b + P_2)}{(b - c)(b - a)}$$

Poniendo z = c y z = a obtenemos q0 y q1

$$q0 = \frac{(a - b)(c^3 + P_1 c^2 + P_2 c) - (c - b)(a^3 + P_1 a^2 + P_2 a)}{K(c - a)(c - b)(a - b)}$$

$$q1 = \frac{c(c - b)(a^3 + P_1 a^2 + P_2 a) - a(a - b)(c^3 + P_1 c^2 + P_2 c)}{K(c - a)(c - b)(a - b)}$$

$$T(z) = A_0 Bm' = A_0 \frac{Bm}{B'} = \frac{(1 + P_1 + P_2) z}{K}$$

$$S(z) = S_1(z)B^+ = z - s1$$

Considerando la señal de control :

$$U(z) = \frac{T(z)}{S(z)} ref - \frac{Q(z)}{S(z)} y$$

Tenemos que :

$$U(z) = \frac{T_0 z\ ref - q_0 z y - q_1 y}{s_0 z - s_1}$$

$$U(z) = \frac{z(T_0\ ref - q_0 y) - q_1 y}{s_0 z - s_1}$$

$$U(z) = \frac{(T_0\ ref - q_0 y) - q_1 y z^{-1}}{s_0 - s_1 z^{-1}}$$

De donde se obtiene :

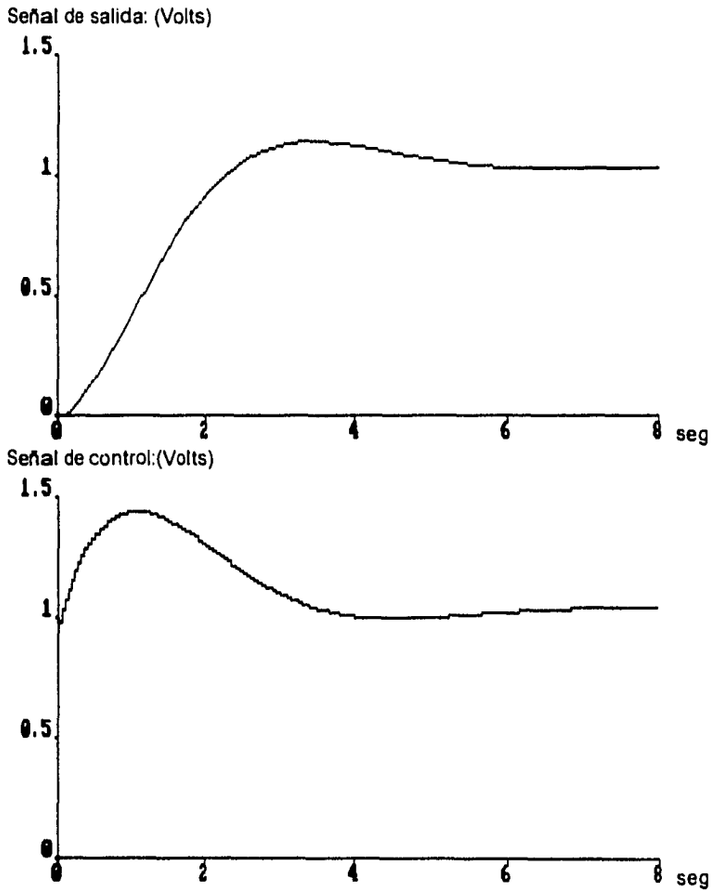
$$\mathbf{U[kh] = t_0 R[kh] - q_0 y[kh] - q_1 y[kh - h] - s1 U[kh - h]}$$

Tomando como ejemplo la misma función de transferencia:

$$Gp(z) = \frac{0.001681(z + 0.96078)}{(z - 0.95122)(z - 0.93239)}$$

aplicando el algoritmo de control obtenido con un sobrepaso de Mp = 0.1 y un Tr = 2.4s se obtiene el siguiente comportamiento para la señal de control y la señal de salida:

(con una frecuencia de muestreo de 20 Hz)



Como se puede observar la señal de control se comporta de una manera menos variante que el algoritmo con cancelación del cero. Por lo tanto, se implementará el algoritmo sin cancelación del cero, ya que con este podemos fijar la referencia del sistema en cualquier rango permisible.

III. EL MICROCONTROLADOR MC68HC11F1

III.1. Características y Diagrama de Bloques

El **MC68HC11F1** es miembro de la familia de microcontroladores MC68HC11. Este circuito contiene funciones sofisticadas en un solo chip. La tecnología CMOS usada en el MC68HC11F1 **(HC = High Density Complementary CMOS)** combina su pequeño tamaño y alta velocidad con el bajo consumo de energía eléctrica y alta inmunidad al ruido.

Las **características principales** del microcontrolador son las siguientes:

- CPU MC68HC11 de 8 bits de datos.
- Alta Velocidad.
- Sistema de Reloj (Timer) de 16 bits.
- Puerto Serial de Comunicaciones Asíncrono (SCI) que utiliza señales en formato NRZ (No retorno a Cero).
- Puerto Serial de Comunicaciones Síncrono (SPI) utilizado como interface de dispositivos externos y otros microprocesadores.
- Convertidor Analógico/Digital de 8 bits, con 8 entradas (canales) multiplexadas.
- 512 Bytes de EEPROM.
- 1024 Bytes de RAM
- 4 Chip Selects Programables.
- Circuito de Interrupciones en Tiempo Real.
- Circuito acumulador de pulsos de 8 bits.
- Sistema de Watchdog COP, el cual se encarga de la operación correcta de la computadora.
- Expansión externa a 64K de Memoria.
- Circuito empaquetado de 68 pines.

En la **figura 3.1** aparece el ***Diagrama de Bloque del MC68HC11F1***.Este diagrama muestra los subsistemas principales y como se relacionan estos con cada uno de los pines del microcontrolador.

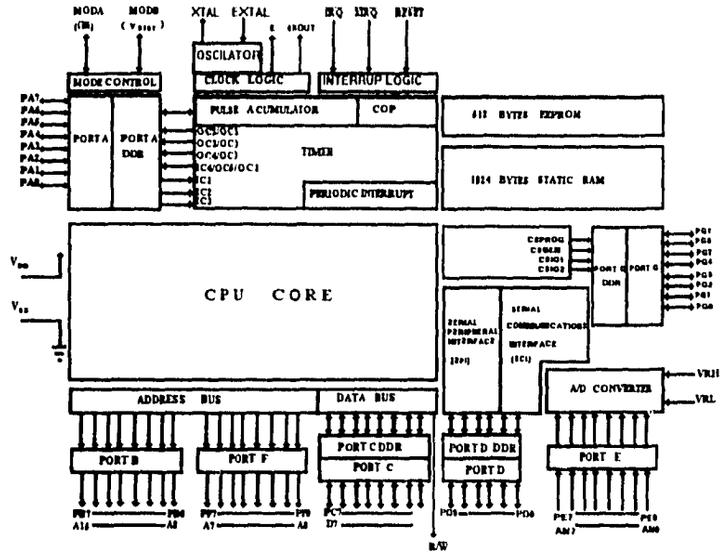


Fig. 3.1. Diagrama de Bloques del MC68HC11F1

III.2. CPU y Modelo de Programación.

III.2.1. CPU

El CPU puede ejecutar todas las instrucciones del M6800y M6801 (Código fuente y Código objeto son compatibles) y las nuevas instrucciones agregadas al MC68HC11F1.

El análisis del CPU interno del MC68HC11F1 se basa en los siguientes elementos:

- Capacidad de 8 bits de datos.
- Modelo de Programación.
    a) Registro de datos
    b) Registro de direcciones
    c) Registro de Estatus
- Set de Instrucciones.

La arquitectura del CPU contiene lo siguiente:

- ALU (Aritmetic Logic Unit).
- Registros Internos: Acumuladores, Apuntadores, Banderas de Estatus.

- Unidad de Control.

- Unidad de Reloj.

- Buses Internos: Datos (8 bits), Direcciones (16 bits).

III.2.2. Modelo de Programación.

El modelo de programación del MC68HC11F1 se muestra a continuación en la **figura 3.2**

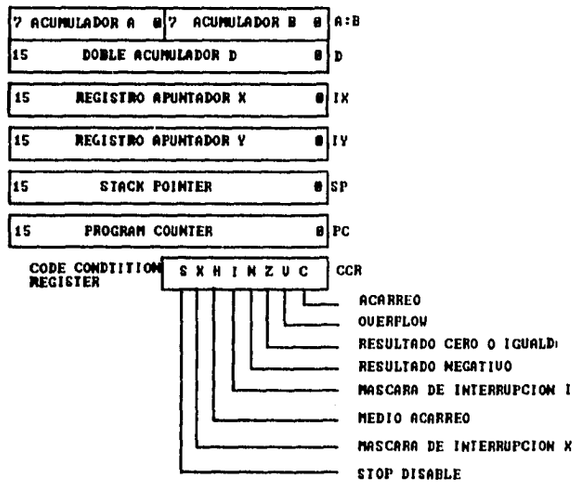


Fig. 3.2. Modelo de Programación

La explicación correspondiente a cada uno de los registros es la siguiente:

- **Registros acumuladores.**

Son 2 registros acumuladores de 8 bits **(A y B)** usados para guardar operandos y resultados de cálculos aritméticos o manipulación de datos. Los acumuladores A y B también se pueden ver como un solo acumulador de 16 bits **(Acumulador D).**

• **Registros apuntadores de índice.**

Son 2 registros apuntadores de índice de 16 bits **(X y Y)** usados para accesar localidades de memoria a través del uso de índices.

• **Registro Contador de Programa.**

El contador de programa **PC (Program Counter)** es un registro de 16 bits en el cual se almacena la siguiente instrucción a ejecutar dentro del microprocesador.

• **Registro apuntador de la Pila.**

El MC68HC11F1 soporta el manejo de una área de **Stack(Pila)**. Este programa se encuentra comúnmente localizado en un espacio de 64Kbytes.

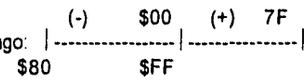
El **SP (Stack Pointer)** es un registro de 16 bits que contiene la dirección del inicio(tope) de la pila y dependiendo de las operaciones que se efectúen sobre ella (Push ó Pop) determinará la posición del SP dentro de la Pila.

• **Registro de Banderas.**

El **CCR (Condition Code Register)** de 8 bits, el cual indica la condición de estado del HC11. **Contiene 5 indicadores de estado, 2 bits de máscara de interrupción y 1 bit de STOP.**

Los Bits **indicadores de estado** son los siguientes:

N: Número Negativo, tomando encuenta el siguiente rango:



Z: Resultado Cero o Igualdad.

V: Overflow (Saturación del Registro)

C: Carry (Acarreo)

H: Medio Acarreo

Los Bits de **Máscara de Interrupción** son:

X: Máscara de Interrupción X, usado para deshabilitar las interrupciones del pin XIRQ

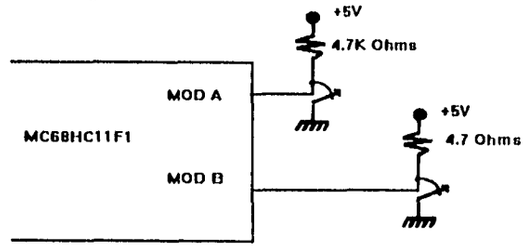
Y: Máscara de Interrupción I, es una máscara global que deshabilita el origen de todas las interrupciones mascarables.

Bit de **STOP disable (S)**, usado para permitir o no permitir la instrucción STOP. Algunos usuarios consideran la instrucción STOP como peligrosa debido a que detiene el oscilador. Este bit también se enciende cuando el sistema detecta alguna falla.

III.3. Modos de Operación.

El **MC68HC11F1** cuenta con cuatro modos de operación: Single Chip, Expandido, Boostrap y Test. Para seleccionar estos modos se utilizan los pines MODA y MODB del HC11 como se muestra en la **figura 3.3**.

Fig. 3.3. Pines del modo de Operación



La selección del modo de operación esta basada en la siguiente tabla.

MODA	MODB	MODO DE OPERACION
0	0	Especial Boostrap
0	1	Single Chip
1	0	Especial Test
1	1	Expandido No-Multiplexado

III.3.1. Modo Single Chip.

Este modo no requiere manejo de direcciones y datos externos, es por eso que se cuenta con más puertos disponibles de propósito general I/O. También en este modo todo el código de programa necesita estar en **memorias internas del MCU (RAM y EEPROM)**.

III.3.2. Modo Expandido No-Multiplexado

El MC68HC11F1 principalmente opera en este modo, debido a que permite conectar memorias y otros dispositivos de manera externa.

En este modo permite realizar una aplicación más completa. Una ventaja de este modo es que el MCU puede direccionar hasta **64KBytes de espacio de direcciones.** Para realizar este direccionamiento se utiliza el **Puerto B (Para los bits de la parte alta de la dirección) y el Puerto F (Para los bits de la parte baja de la dirección)** . El **Puerto C se utiliza para la transferencia de los datos.**

El pin de R/W es usado para controlar la salida de datos en el Puerto C, esto se encarga de coordinar al microcontrolador cuando existe una operación de lectura o una de escritura. Además en este modo se cuenta con 4 Chip Selects Disponibles en lós pines PG7-PG4 del Puerto G, los cuales pueden ser utilizados como líneas de control externas.

III.3.3. Modo Especial Boostrap.

El modo Boostrap es similar a Single Chip. Después del Reset , en este modo se habilita una ROM en las direcciones: $BF00-$BFFF. El vector de Reset es buscado en las direcciones: $BFC0-$BFFF. Posteriormente el MCU procede a ejecutar las rutinas encontradas en esta ROM.

Estas rutinas inicializan el SCI del sistema, revisan las opciones de seguridad y aceptan el envío de programas a través del SCI, estos programas son cargados en la RAM y finalmente salta a la

dirección $0000 (en la RAM) para empezar a ejecutarse el código contenido a partir de esta dirección. Eso permite que los programas puedan ser cargados y ejecutados al momento de iniciar el sistema.

En modo Boostrap todos los vectores de interrupción son mapeados en la RAM. La Tabla de vectores de Interrupción es la siguiente:

DIRECCION	VECTOR
00C4	SCI
00C7	SPI
00CA	Pulso Acumulador de entrada
00CD	Pulso Acumulador de Overflow
00D0	Overflow
00D3	Salida de comparación 5
00D6	Salida de comparación 4
00D9	Salida de comparación 3
00DC	Salida de comparación 2
00DF	Salida de comparación 1
00E2	Entrada de captura 3
00E5	Entrada de captura 2
00E8	Entrada de captura 1
00EB	Interrupción de Tiempo-Real
00EE	IRQ
00F1	XIRQ
00F4	SWI
00F7	Código Ilegal
00FD	Reloj de Monitoreo
00FF	Reset

III.3.4. Modo Especial Test.

El objetivo principal de este modo es la calibración de programas o personalizar datos dentro de la EEPROM. En general este modo es usado para revisar las fases de un proyecto antes de ponerlo en funcionamiento. La EEPROM es desactivada en este modo. Uno como usuario puede hacer diferentes pruebas control de bits.

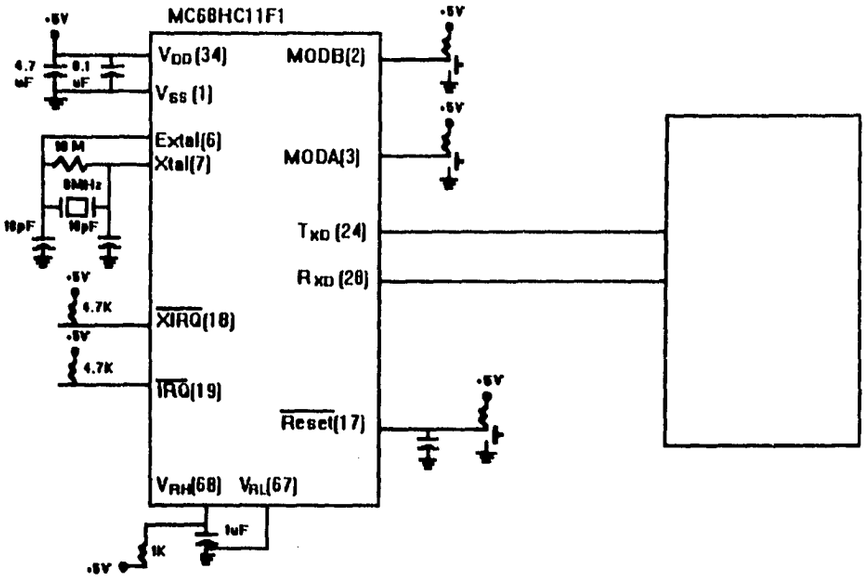
Los vectores de Reset e Interrupción son buscados externamente en las direcciones:

$BFC0-$BFFF

III.4. CONFIGURACION EN MODO EXPANDIDO UTILIZADA PARA EL CONTROLADOR DIGITAL.

En la **figura 3.4** aparece la configuración en modo expandido utilizada para el controlador digital. Dicha configuración esta diseñada de tal forma que el sistema sea fácil de modificar.

figura 3.4



III.4.1. Descripción de señales.

III.4.1.1. VDD y VSS

VDD es el voltaje de polarización del HC11, los valores que maneja son: **+5V ±10%**

VSS representa la tierra física, la cual tiene un valor de **0V.**

Entre los pines de **VDD Y VSS** se encuentra conectado un capacitor de **0.01µF (Capacitor de Bypass)** el cual asegura frecuencias altas en el micro. Esto se usa debido a que algunas veces se presenta en los pines transiciones de señales de manera muy rápida.

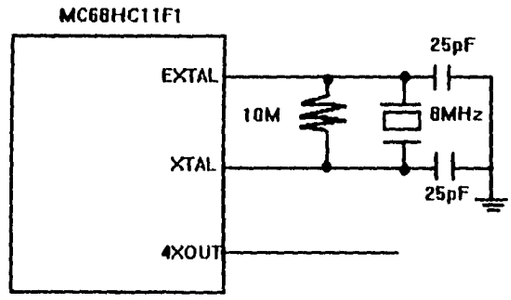
III.4.1.2. Reset

Es una señal de control bidireccional (activa en bajo). Esta señal es usada para **inicializar el sistema** y usada como salida que indica que se detectó una falla en el micro en el monitoreo de **reloj en el circuito Watchdog.**

Iii.4.1.3. XTAL Y EXTAL

Estos pines proveen la interface de reloj utilizando para ello un cristal(timer) CMOS, a través del cual se alimenta el reloj interno del HC11 (la frecuencia de este reloj **(Reloj E)** es la cuarta parte de la frecuencia del cristal utilizado) y se genera un reloj externo que puede ser utilizado para comunicación con otros dispositivos externos.

La configuración típica es la que se muestra en la **figura 3.5**



III.4.1.4 VRH y VRL

Estos voltajes fijan la referencia que usará el convertidor analógico digital, que es un convertidor de 8 bits.

IV.- REALIZACIÓN DEL SISTEMA

IV.- REALIZACION DEL SISTEMA

En este capítulo se describe todo el diseño del controlador digital desarrollado. Dentro del hardware se describen los componentes utilizados y las configuraciones usadas de estos. Dentro de la parte del software se describe los algoritmos desarrollados así como las bibliotecas utilizadas para el desarrollo de los mismos.

IV.1- Desarrollo del Hardware

En cuanto el hardware, se logró que el número de componentes para el desarrollo del trabajo en cuestión, no fuera muy extenso, gracias a la multitud de características que nos proporcionó el microcontrolador HC11, como es el convertidor analógico/digital de 8 bits, la posibilidad de manejar memoria expandida, y los puertos multiusos, como se describe en el capítulo anterior.

En lo que se refiere a los periféricos usados, para implementar el teclado se usaron tres interruptores de tipo pushbutton, con los cuales se puede configurar de manera local el controlador, también se usó una pantalla de cristal líquido que nos permite desplegar texto y gráfico, y éste nos sirve para observar el comportamiento del sistema a controlar mediante una gráfica, ver los valores actuales de los parámetros de los algoritmos de control, así como la selección del algoritmo de control.

Como el dispositivo maneja un rango de voltaje de +- 10V , se tuvieron que implementar etapas de acoplamiento las cuales se hicieron mediante amplificadores operacionales.

Para generar la salida de control, que consiste en una señal analógica, que se obtiene a partir del algoritmo de control, se requirió usar el convertidor digital/analógico de 8 bits para obtener dicha señal. Todos estos componentes se muestran en la diagrama de bloques de la figura 1.2.

En los siguientes puntos se detallará con mayor precisión el desarrollo de cada una de las partes descritas anteriormente.

IV.1.1- Interface de Operación de Teclado

Como se había descrito anteriormente el teclado de operación del sistema de control está constituido por tres teclas implementadas por tres pushbutton, cada una de estas teclas fueron conectadas de la siguiente manera:

Tecla de Incremento	Puerto E bit 7
Tecla de Aceptar	Puerto E bit 6
Tecla de Decremento	Puerto E bit 5

Estas teclas fueron conectadas de tal manera que cuando se oprima alguna de ellas proporcione al respectivo bit del puerto E una señal lógica de cero (0 Volts), y cuando no se esté presionando ninguna de ellas la señal de entrada sea de un uno lógico (5 Volts), de esta manera mediante una constante de verificación del estado del puerto E podremos saber si se encuentra presionada alguna tecla y tomar la acción correspondiente.

La configuración física se muestra le figura 4.1

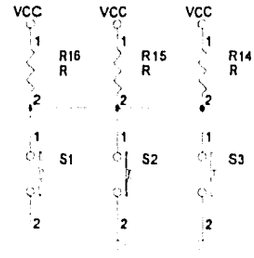


figura 4.1

Como se puede observar, cada tecla tiene conectada en un extremo una resistencia y esta resistencia va conectada a Vcc, el otro extremo de la tecla es conectada a tierra, la función de la resistencia es, para que cuando se presione la tecla no se provoque un corto circuito y obtengamos el valor binario deseado.

IV.1.2 Interface de Despliegue

La interface de despliegue está implementada por una pantalla de cristal líquido la cual nos permite visualizar tanto texto, como gráficos. El modelo usado es el AND1021ST, que es un dispositivo con las siguientes características:

En modo texto : 15 caracteres por 8 líneas

En modo gráfico: 120 x 64 pixeles

Está construido con el controlador de display de cristal líquido T6963C

Rango de temperatura de operación 0 a 50°C

Estas características son suficientes para lograr una buena visualización de la gráfica de respuesta del sistema, así como los datos involucrados en este.

El control de este display se hace mediante un conjunto de pins los cuales se describen a continuación:

Pin 1 FGND (Tierra física)	Pin 10 RESET
Pin 2 GND (Tierra 0 volts)	Pin 11 D0 (Data I/O)
Pin 3 Vdd (5 volts)	Pin 12 D1 (Data I/O)
Pin 4 Vee (-5.5 V a -11.5 V)	Pin 13 D2 (Data I/O)
Pin 5 WR (Data Write)	Pin 14 D3 (Data I/O)
Pin 6 RD (Data Read)	Pin 15 D4 (Data I/O)
Pin 7 CE (Chip Enable)	Pin 16 D5 (Data I/O)
Pin 8 C/D (Command/Data)	Pin 17 D6 (Data I/O)
Pin 9 NC (No Connection)	Pin 18 D7 (Data I/O)
	Pin 19 y Pin 20 (No Connection)

Empezaremos describiendo los pines de alimentación como es el Pin 2 y el Pin 3 los cuales se conectan a tierra y a Vcc directamente, el Pin 4 nos sirve para fijar la intensidad del despliegue en la pantalla, este voltaje va relacionado con la temperatura ambiental y dependiendo de esta se puede ajustar hasta lograr una buena visualización de los datos, en este caso, nosotros fijamos el voltaje a -8.0 V, para obtener este voltaje y considerando que estamos manejando para todo el

sistema una fuente de alimentación de +- 5 V y +- 10 V , utilizamos un regulador de voltaje, que fija el voltaje a -8.0 V, este dispositivo es el 7908 cuya configuración se muestra en la figura 4.2.

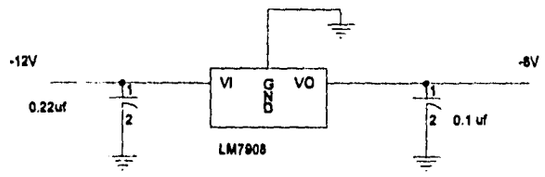


figura 4.2

Los pines de datos (pin 11 al Pin 18) se conectaron a la salida de un *latch* de 8 bits para efectos de temporización, y con esto lograr mandar datos al display, las entradas de dicho *latch* son conectados a las líneas de datos del microcontrolador (Puerto C), el pin de habilitación del *latch* fue conectado al pin del controlador CSI01 (Chip Select I/O 1) por lo cual para mandar un dato al display hay que accesar a alguna dirección de memoria ubicada dentro de $1060-$17FF que es el rango asignado a este chip select para dispositivos externos que nos proporciona el microcontrolador.

Por último tenemos cuatro pines de control de display que son WR (Write), RD (Read), CE (Chip enable) y C/D (Command / Data), que deben ser controlados o activados de acuerdo el diagrama de tiempos proporcionado por el manual que describe el display (AND LED and LCD Products), para lograr este objetivo estos pines se conectaron al puerto G del pin 0 al pin 3, con lo cual tenemos un absoluto control de estas señales, escribiendo en el puerto los valores correspondientes. A continuación se muestra la conexión entre los pines del display y el HC11.

Pin RD	Puerto G bit 0
Pin CE	Puerto G bit 1
Pin WR	Puerto G bit 2
Pin C/D	Puerto G bit 3

IV.1.3 Circuitos de acondicionamiento de señales.

IV.1.3.1 Acondicionamiento de la señal de entrada.

Como se menciona al principio de este capítulo, nuestro sistema maneja un rango de +-10 V de entrada y salida, para la señal de la planta y la señal de control respectivamente, por lo cual fue necesario implementar circuitos de acondicionamiento para que estas señales fueran operables por el microcontrolador, ya que el convertidor analógico /digital del microcontrolador acepta un entrada en el rango de 0 a 5 V.

A la entrada primero usamos un filtro paso bajas activo con ganancia de 0.25, es decir reducirá la señal cuatro veces, este filtro con frecuencia de corte de 200 Hz es utilizado, aparte de atenuar la señal, para eliminar el ruido, una vez que la señal pasó por el filtro su rango será de +-2.5 V y estará invertida, con lo cual ya tenemos una señal de máximo 5 V, a ésta hay que sumarle un constante de DC con un valor de 2.5 V para que esté dentro del rango de 0 a 5 V

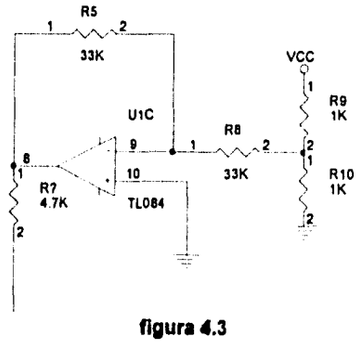


figura 4.3

Mediante la configuración mostrada en la figura 4.3 se obtiene una señal de -2.5 V es negativa porque hay que recordar que la señal que estamos procesando fue invertida por el filtro paso bajas, una vez obtenidas estas dos señales las sumamos con un amplificador operacional, el cual sumará e invertirá el resultado, con lo que obtendremos finalmente una señal en el rango de operación del microcontrolador, de 0 a 5V.

Cálculo del Filtro paso bajas de segundo orden.

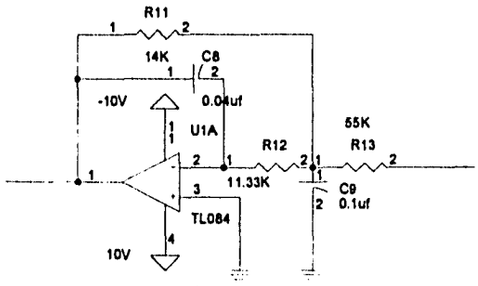


figura 4.4

La configuración de un filtro paso bajas activo se muestra en la figura 4.4 tenemos que calcular los capacitores y resistencias en base a los siguientes datos:

Ganancia Ho = 0.25
Calidad Q = 0.7071 (para un filtro de segundo orden)
Frecuencia de corte Fc = 200 Hz

Se fija el valor del capacitor C2 = 0.1uF

Cálculo de la constante m

$m=1 / (4 \times Q^{2} \times (Ho+1))$ Haciendo los respectivos cálculos tenemos que m=0.4

Determinación de C5,R4,R1,R3 :

C5 = m x C2	Haciendo el cálculo	C5=0.04uF
R4 = 1 / (2 x Q x Wc x C2 x m)	Haciendo el cálculo	R4=14K
R1 = R4 / Ho	Haciendo el cálculo	R1=56.270K
R3 = R4 / (Ho + 1)	Haciendo el cálculo	R3=11.25K

Generación de la señal de -2.5V

Para generar la señal de 2.5V usamos dos resistencias iguales conectadas en serie y los extremos a 5V y 0V respectivamente, por lo que entre las dos resistencias tendremos aproximadamente 2.5V, para que este voltaje no varíe se conecta en este punto una resistencia con un valor grande, el valor elegido fue de 33K , en su otro extremo se conectará a un amplificador operacional configurado como inversor de ganancia unitaria, con lo cual obtendremos la señal de -2.5V, como se muestra en la figura 4.3.

Sumador

Finalmente tenemos un sumador implementado con un amplificador operacional de ganancia unitaria, con el sumamos la constante de -2.5V y la señal de salida del filtro paso bajas, para obtener la señal en el rango de 0 a 5V como se había mencionado. La configuración del sumador se muestra en la figura 4.5.

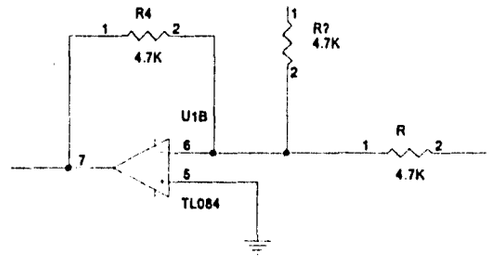


figura 4.5

IV.1.3.2 Acondicionamiento de la señal de salida

La señal de control que debe ser generada por nuestro sistema debe estar dentro del rango de +-10V por lo tanto también se requiere un circuito analógico que nos proporcione dichos voltajes.

La señal de control digital tiene una precisión de 8 bits que salen del puerto A del microcontrolador, estos son convertidos a la respectiva señal analógica dentro del de rango de 0 a 5V mediante el convertidor digital/analógico de 8 bits DAC0808.

Después a esta señal le sumamos un valor constante de -2.5V de DC, que se obtienen de manera similar a la utilizada en el circuito de acondicionamiento de entrada, es decir conectamos dos resistencias iguales en serie y los extremos a -5V y 0V respectivamente y en el punto intermedio una resistencia de un valor grande, en este caso de 56K. Esta señal constante y la señal proveniente del convertidor digital/analógico son sumadas mediante un amplificador operacional configurado como sumador de ganancia unitaria, cuya configuración se muestra en la figura 4.6, y

con esto ya tenemos una señal en el rango de +-2.5V.

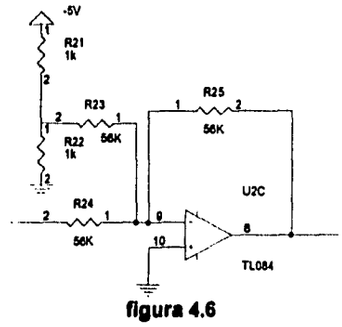


figura 4.6

Por último hay que amplificar la señal obtenida 4 veces, por lo tanto utilizamos un amplificador con ganancia de 4 y obtenemos una señal el rango de +-10V. La configuración de dicho amplificador se muestra en la figura 4.7.

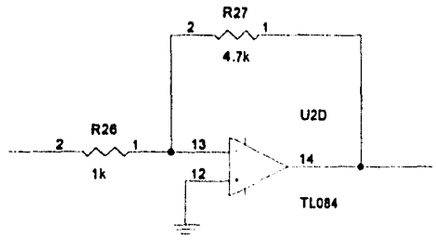


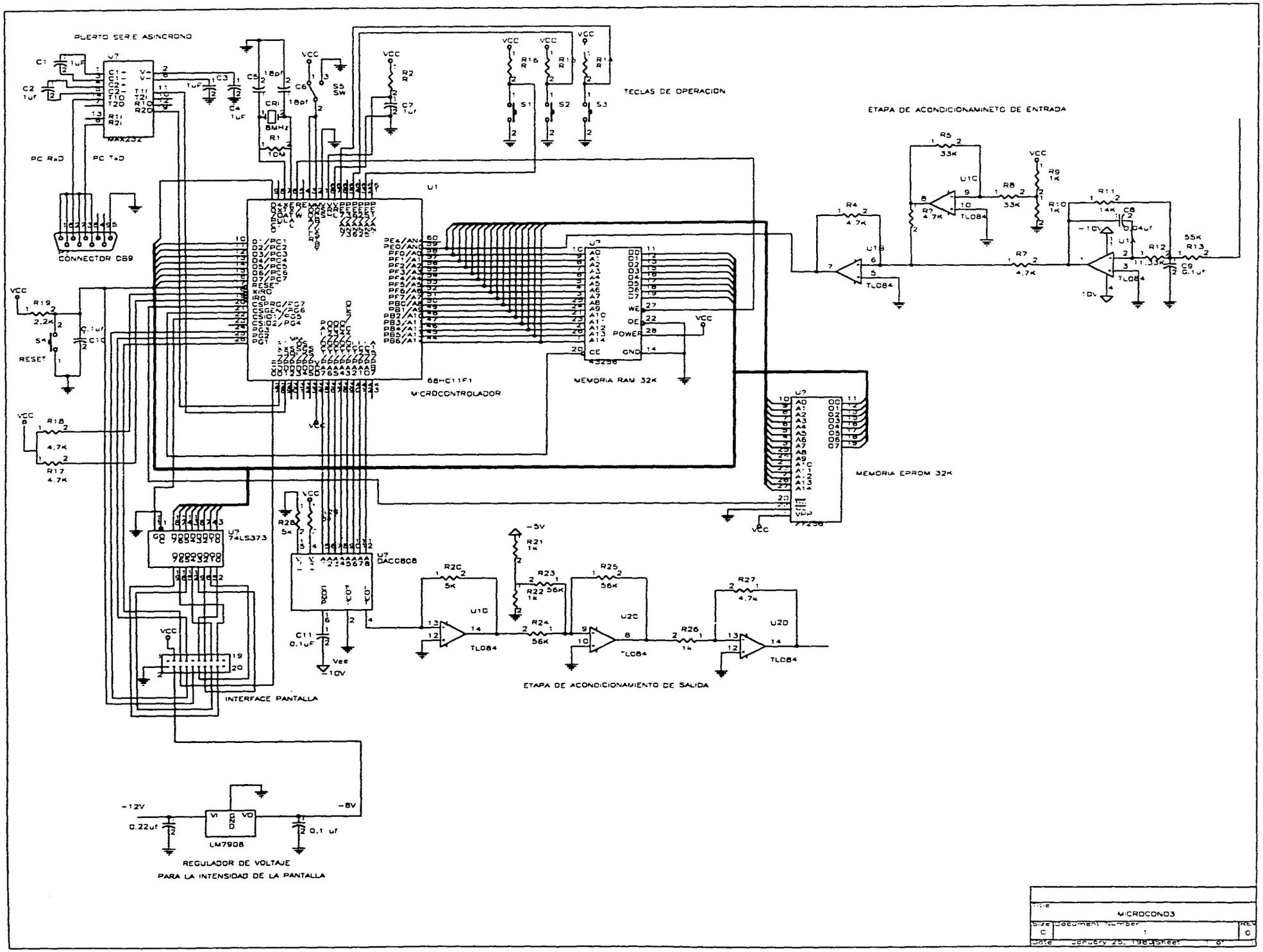
figura 4.7

IV.1.4 Diagrama General

A continuación se muestra el diagrama general con todos los componentes del dispositivo, descritos anteriormente, así como la memoria externa e interface del dispositivo con el puerto serial de una computadora personal para poder transferir los programas de operación.

El microcontrolador es configurado en modo expandido por lo tanto usa memoria externa, que en nuestro caso se trata de una memoria RAM de 32 Kbytes, la cual se conecta a las respectivas líneas de datos y dirección del microcontrolador.

Mediante el circuito MAX232 se implemento la interface del este dispositivo con una computadora personal, esto es con el fin de acoplar la señal de +-12V del puerto serial de la PC con el puerto serie del microcontrolador que maneja un rango de 0V a 5V.


PUERTO SERIE ASINCRONO
MAX232
CONNECTOR DB9
PC RxD
PC TxD
TECLAS DE OPERACION
S1
S2
S3
S5
SW
RESET
68HC11F1
MICROCONTROLADOR
MEMORIA RAM 32K
MEMORIA EPROM 32K
ETAPA DE ACONDICIONAMINETO DE ENTRADA
TL084
ETAPA DE ACONDICIONAMIENTO DE SALIDA
DAC0808
74LS373
INTERFACE PANTALLA
LM7908
REGULADOR DE VOLTAJE
PARA LA INTENSIDAD DE LA PANTALLA
VCC
-5V
-10V
-12V
-8V
Vee
MICROCONO3

IV.2 Desarrollo de Software

En esta parte se describirá detalladamente todo el software desarrollado para el funcionamiento del sistema, explicaremos como está estructurado, que librerías fueron utilizadas y el flujo del programa.

Todos los programas fueron desarrollados en lenguaje ensamblador 68HC11 y fueron compilados mediante el compilador y editor IASM11, el software fue cargado en las memorias del microcontrolador, a través de una computadora personal y mediante un software especial para comunicarse con el microcontrolador vía puerto serie de una manera asíncrona, dicho software es el PCBUG11.

IV.2.1 Librerías de lenguaje máquina

IV.2.1.1 Librerías ya existentes.

Los algoritmos de control utilizados requieren cálculos y operaciones en punto flotante, el lenguaje ensamblador no incluye este tipo de operaciones por lo tanto, hay que programarlas. Los algoritmos para efectuar operaciones de punto flotante son muy complejos y grandes, por lo tanto, se utilizaron algoritmos ya implementados en los microcontroladores de la familia HC11, estos algoritmos son provistos en las notas de aplicación de Motorola "MC68HC11 Floating-Point Package"

Este paquete contiene además de las cuatro operaciones básicas (suma, resta, multiplicación y división), rutinas de conversión de ASCII a punto flotante y viceversa, también contiene tres funciones trigonométricas básicas seno, coseno y tangente con rutinas de conversión de grados a radianes y de radianes a grados, así como también la raíz cuadrada.

Como en muchas aplicaciones se utilizan por lo regular números enteros este paquete también tiene rutinas de conversión de entero a flotante y de flotante a entero.

Estas rutinas ocupan alrededor de 3K bytes en memoria y sólo requiere 10 bytes en memoria RAM y el stack en RAM. Todas las variables temporales residen en el stack, esto ayuda a que no exista interferencia con las rutinas de interrupción.

El formato usado para almacenar números en las rutinas de punto flotante consiste en diez bytes en RAM los cuales son usados para dos acumuladores FPACC1 y FPACC2. Cada acumulador de cinco bytes de tamaño, consiste en un byte para el exponente, tres bytes para la mantisa, y un byte para indicar el signo de la mantisa.

El byte del exponente es usado para indicar la posición del punto binario y es por el decimal 128 ($80) para que las comparaciones sean mas fáciles. Este byte proporciona un rango dinámico de 1x10+-38.

La mantisa consiste de tres bytes (24 bits) y es usado para almacenar la parte entera y la parte fraccional del número en punto flotante. Estos 24 bits proveen mas de siete dígitos de precisión.

El último byte es usado para indicar el signo de la mantisa, cuando la mantisa es positiva este es $00. Cuando tenemos una mantisa negativa tenemos un valor de $FF.

Siete condiciones de error pueden ser detectadas por los algoritmos. Cuando ocurre un error el algoritmo regresa el código correspondiente, los códigos se describen a continuación:

Error	Descripción
1	Error en formato en la conversión de ASCII a flotante
2	Overflow
3	Underflow
4	División entre cero
5	Número flotante muy largo o pequeño para convertirlo a entero
6	Raiz cuadrada de un número negativo
7	Tangente de PI/2

A continuación se describen brevemente como se utiliza algunas de las rutinas que provee el paquete:

Conversión de ASCII a flotante
Nombre: ASCFLT
Operación: ASCII(X) -> FPACC1
Entrada: El registro X apunta a la cadena de caracteres a convertir.
Salida: FPACC1

Multiplicación en punto flotante
Nombre: FLTMUL
Operación: FPACC1xFPACC2 -> FPACC1
Entrada: FPACC1 y FPACC2 contienen los números a multiplicar.
Salida: FPACC1 contiene el producto de los dos acumuladores.

Suma en punto flotante
Nombre: FLTADD
Operación: FPACC1 + FPACC2 -> FPACC1
Entrada: FPACC1 y FPACC2 contienen los sumandos.
Salida: FPACC1 contiene la suma de los acumuladores.

Resta en punto flotante
Nombre: FLTSUB
Operación: FPACC1 - FPACC2 -> FPACC1
Entrada: FPACC1 y FPACC2 contienen los sumandos.
Salida: FPACC1 contiene la suma de los acumuladores.

División en punto flotante
Nombre: FLTDIV
Operación: FPACC1 / FPACC2 -> FPACC1
Entrada: FPACC1 y FPACC2 contienen el divisor y el dividendo respectivamente
Salida: FPACC1 contiene el cociente.

Conversión de punto flotante a ASCII
Nombre: FLTASC
Operación: FPACC1 -> (X)
Entrada: FPACC1 contiene el número a convertir a ASCII
Salida: A partir de la dirección apuntada por IX

Comparación en punto flotante
Nombre: FLTCMP
Operación: FPACC1 - FPACC2
Entrada: FPACC1 y FPACC2 contiene los números a comparar.
Salida: Las banderas del microcontrolador son adecuadamente cambiadas.

Conversión de un entero sin signo a flotante
Nombre: UINT2FLT
Operación: (16 bit) -> FPACC1
Entrada: En los 16 bits mas bajos de la mantisa de FPACC1
Salida: FPACC1

Conversión de flotante a entero
Nombre: FLT2INT
Operación: FPACC1 -> (16 bit)
Entrada: FPACC1 puede contener un número flotante entre 65535 y -32767
Salida: En los 16 bits mas bajos de la mantisa de FPACC1.

Transferencia de FPACC1 a FPACC2
Nombre: TFR1TO2
Operación: FPACC1 -> FPACC2
Entrada: FPACC1
Salida: FPACC2

Las funciones en punto flotante son calculadas mediante series de Taylor. En seguida se describe el uso de algunas de ellas:

Raíz cuadrada
Nombre: FLTSQR
Operación: Calcula la raíz cuadrada de un flotante.
Entrada: FPACC1
Salida: FPACC1

Seno
Nombre: FLTSIN
Operación: SIN(FPACC1) -> FPACC1
Entrada: FPACC1 contiene el ángulo en radianes en el rango de -2PI a 2PI.
Salida: FPACC1

Coseno
Nombre: FLTCOS
Operación: COS(FPACC1) -> FPACC1
Entrada: FPACC1 contiene el ángulo en radianes en el rango de -2PI a 2PI.
Salida: FPACC1

IV.2.1.2 Librerías implementadas

Como faltaban algunas funciones de punto flotante como logaritmo natural, exponencial y ángulo coseno, y dichas operaciones son requeridos en los algoritmos de control utilizados en el sistema, se implementaron estas siguiendo la estructura de las rutinas del punto flotante hechas por Motorola, se utilizaron también series de Taylor y se fijó el número de ciclos adecuado para evitar errores, pero sin disminuir mucho el tiempo de respuesta del algoritmo de control. A continuación se describen estas tres funciones para números en punto flotante.

Logaritmo natural.- Como ya se había dicho se usaron series de Taylor para implementarla la serie es la siguiente:

$\ln (X) = (X - 1) - (X - 1)^2 / 2 + (X - 3)^3 / 3 - (X - 4)^4 / 4$ .....

El nombre de la subrutina es FLTLOG

El dato se manda en FPACC1 y el resultado es puesto en FPACC1

Exponencial.- La serie de Taylor programada para sacar el exponencial de un número es el siguiente:

$e \wedge (X) = 1 + X + X^2 / 2! + X^3 / 3! + X^4 / 4!$ .....

El nombre de la subrutina es EXPON

El dato se manda en la variable EXPONENTE y el resultado es puesto en FPACC1

Ángulo coseno.- La serie de Taylor programada para sacar el ángulo coseno de un número es el siguiente:

$\cos \wedge{-1} (X) = PI / 2 - (X + 1/2 \times X^3 / 3 + (1x3 / 2x3) \times X^5 / 5 + (1x3x5 / 2x4x6) \times X^7 / 7)$

El nombre de la subrutina es FLTANGCOS

El dato es mandado en FPACC1 y el resultado es puesto en FPACC1

IV.2.2 Programación de Periféricos

En esta sección, se describirá detalladamente en que consisten las rutinas implementadas para operar correctamente los diferentes periféricos externos al microcontrolador, utilizados en el sistema realizado, como es la pantalla, el teclado y el convertidor digital/analógico de ocho bits.

IV.2.2.1 Operación de la pantalla de despliegue

Como se había mencionado anteriormente, la pantalla utilizada en el sistema nos permite visualizar datos en modo gráfico, en modo texto o la combinación de los dos. Para controlar el display se tiene que seguir con cierto diagrama de tiempos, para habilitar el display, escribir un comando o dato, este diagrama de tiempos se puede consultar en la respectiva referencia.

Para la operación del display se necesitan básicamente dos tipos de valores a mandar en forma paralela, estos son datos ó comandos. Los datos son valores predeterminados que nos pueden indicar, dependiendo del comando que se esté realizando, direcciones, tamaños de memoria, el código de algún carácter o los pixeles a prender en el caso de que estemos trabajando en modo gráfico. Por otro lado los comandos nos sirven para indicarle al display la función a realizar, como pueden ser: modo de despliegue, tipo de cursor, dirección de memoria a apuntar, dirección en donde empieza el texto, etc.

Para indicarle al display si se trata de un comando o un dato y de acuerdo al diagrama de tiempos correspondiente, se programaron las dos subrutinas que se muestran en seguida:

```
COMANDO EQU *
    LDAA #$09     ; Máscara 1001 Activa WR y CE (negativos PG3 y PG2).
    STAA PORTG
    STAB PORTC
    LDAA #$0F     ; Máscara 1111 Desactiva los bits activados.
    STAA PORTG
    RTS
```

En la rutina para mandar un comando al display, que se muestra anteriormente, el código del comando debe ser previamente cargado en el registro B del microcontrolador, el cual será

puesto en la dirección que corresponde a PORTC, que es en realidad la dirección que corresponde al chip select CSIO1, el cual se activará en ese momento. Accesando al puerto G con los datos correspondientes se activa el write, el chip enable y se le indicará al display que se trata de un comando.

De la misma manera se opera en la rutina mostrada a continuación, pero en este caso mediante el dato escrito en el puerto G se le indica al display que se trata de un dato.

```
DATO   EQU *
       LDAA #$01      ; Máscara 0001 Activa D,WR y CE (negativos).
       STAA PORTG
       STAB PORTC
       LDAA #$07      ; Máscara 0111 Desactiva WR y CE.
       STAA PORTG
       RTS
```

Antes de comenzar a utilizar el display se necesita mandarle una serie de comandos y datos para inicializarlo, dicha secuencia se muestra a continuación:

- Indicación del modo de operación que puede ser "OR", "EXOR", "AND" y capaz de tener atributo.

- Dirección de memoria donde empieza el texto, esta depende del modelo del display para el AND1021 es la $1000.

- Tamaño del área de texto, esta también depende del modelo en nuestro caso es $0F.

- Dirección de memoria donde empieza el área gráfica, en nuestro caso es $0000.

- Tamaño del área gráfica, para nuestro caso es $0F.

Con lo anterior ya está listo el display para desplegar datos de tipo gráfico y texto. Para una manejo más fácil del display se implementaron diferentes subrutinas, como por ejemplo, borrar la pantalla en modo texto, borrar la pantalla en modo gráfico, dibujar un pixel en ciertas coordenadas X, Y, posicionar el cursor en X, Y, y escribir una cadena de texto. A continuación se describe cada una de ellas.

Borrado de la pantalla en modo texto.- En esta subrutina lo que se hace es escribir en toda el área de texto caracteres nulos, con esto logramos borrar lo que anteriormente aparecía. El nombre de la subrutina es CLRTXT y se puede consultar en el código fuente del programa.

Borrado de la pantalla en modo gráfico.- Igual que la subrutina anterior en esta se escriben ceros en las direcciones de memoria correspondientes al modo gráfico. El nombre de la subrutina es CLRGRAPH .

Posicionar el cursor en las coordenadas X ,Y.- Esta subrutina mediante direcciones predeterminadas que corresponden a cada uno de los renglones en modo texto, se le suma un desplazamiento para obtener la deseada posición X, Y. El nombre de esta subrutina es GOTOXY y la posición se manda en la variable WHERE.

Escribir un pixel en las coordenadas X, Y.- Mediante esta subrutina se escribe un pixel en modo gráfico en la posición X, Y, esta subrutina es un poco mas compleja ya que cada byte de la zona gráfica tiene la información de 8 pixeles, y por lo tanto para no modificar algún pixel que ya esté activado hay que comparar con máscaras, calcular bit correspondiente en el byte y sólo prender este. El nombre de la subrutina es PUTXY, y las coordenadas se deben escribir previamente en las variables "codx y cody".

Escribir una cadena de texto.- Esta es una subrutina que escribe cadenas de caracteres de diferente tamaño, y como lo hacen algunos lenguajes de alto nivel para saber donde termina la cadena se pone un indicador que usualmente es el número hexadecimal $00. El código de caracteres manejado por el display es similar al código ASCII, pero con cierto desplazamiento, por lo cual antes de escribir alguna cadena de caracteres hay que convertirla al código de caracteres del display, esta operación es efectuada por la subrutina llamada CONVIERTE. La subrutina que escribe la cadena de caracteres en el display se llama PRINTF, la dirección de inicio de la cadena es mandada en el registro IX del microcontrolador y se empieza a escribir en la posición actual del cursor.

IV.2.2.2 Operación de las teclas.

Como ya se habla mencionado, el sistema maneja solamente tres teclas con las cuales se puede hacer varias funciones, como por ejemplo activar determinado algoritmo de control, cambiar el valor de algún parámetro de los algoritmos de control, así como el de parar el algoritmo de control, estas teclas están conectadas al puerto E, por lo tanto debemos estar leyendo constantemente este puerto para detectar si se ha oprimido alguna tecla, esta lectura se hace una vez en cada ciclo de control, lo cual es una velocidad adecuada y totalmente transparente al usuario.

La lógica de función de las teclas es muy sencilla, cuando estamos en el menú principal, que es la parte que aparece cuando prendemos el sistema, podemos elegir el algoritmo de control deseado, mediante números los cuales se incrementan o se decrementan con las teclas correspondientes. Una vez que tenemos el número de la opción deseada, sólo hay que oprimir la tecla de aceptar y se activará automáticamente el algoritmo de control seleccionado. Ya dentro de cierto algoritmo de control las teclas de incremento y de decremento nos servirán para ir mostrando cada una de las variables o parámetros del respectivo algoritmo de control. Si el usuario quiere cambiar el valor de alguno de estos parámetros deberá mostrar el parámetro y oprimir aceptar, en este momento las teclas de incremento y de decremento servirán para incrementar o decrementar el valor de la variable, una vez que ya tengamos el valor deseado de la variable sólo hay que oprimir aceptar para fijar este valor, en este momento las teclas de incremento y de decremento regresarán a su función anterior que es la de mostrar cada una de las variables. Una de estas variables tiene el nombre de STOP cuando estamos en esta variable y oprimimos aceptar, paramos el algoritmo de control y regresamos al menú principal.

Cuando estamos en el menú principal se activa un ciclo constante que llama a la subrutina DETECTA la cual va verificar si se ha oprimido alguna tecla, leyendo los bits correspondientes del puerto E. Se saldrá de este ciclo hasta que se haya oprimido aceptar, es decir, hasta que se haya elegido la activación de algún algoritmo de control.

Una vez estando en algún algoritmo de control, al final de cada ciclo de control se llama a la subrutina DETECTA la cual dependiendo de la tecla oprimida llamará a la subrutina correspondiente, que puede ser FLECHAD, FLECHAI ó ACEPTAR. En las subrutina FLECHAD y FLECHAI y se verificará si anteriormente se ha oprimido un aceptar con el objetivo de saber que función debe realizar, si no se ha oprimido aceptar anteriormente, estas rutinas deberán incrementar o decrementar el apuntador a la variable del algoritmo de control a la cual está apuntando, es decir, el apuntador a las variables (variable APUNVAR) que contiene la dirección de alguna de ellas del respectivo algoritmo, apuntará a otra variable del mismo algoritmo, esto, reflejado en el display, es como elegir otra variable. Pero cuando hemos detectado que anteriormente se oprimió aceptar deberemos incrementar o decrementar el valor de la variable a la cual estamos apuntando, esto se hace en las subrutinas TEDERECHA y TEIZQUIERDA, y cuando volvemos a oprimir aceptar, borramos la bandera que nos indica que se ha oprimido aceptar (BANENT) y las teclas de incremento y decremento regresan a su función anterior.

Las subrutinas rutinas VARIABLE, DESPINC y DESPDEC, nos sirven para saber a qué variables estamos apuntando, para incrementar el apuntador a estas variables y para decrementar el apuntador a estas variables, respectivamente.

IV.2.2.3 Convertidor digital/analógico

El convertidor digital/analógico usado es el DAC0808, que es convertidor de 8 bits, una vez que el algoritmo de control en uso ha calculado la señal de control, esta se convierte a número entero, este número entero se escribe en el puerto A, que es un puerto de 8 bits y está conectado directamente a dicho convertidor.

En cada ciclo de control es generada una señal de control, por lo cual el puerto A cambia constantemente de valor y con esto el valor proporcionado por el convertidor digital/analógico.

Un valor entero de 8 bits está dentro del rango de 0 a 255 si consideramos que es sin

signo, pero hay que recordar que nuestro algoritmo de control generará una señal de control en punto flotante en el rango de 0 a 5, por lo cual hay que obtener su valor correspondiente dentro del rango de 0 a 255, entonces una vez que tenemos la señal de control se multiplica por una el valor constante 51 que es la división de 255 / 5. Todo este proceso se realiza en la subrutina SALIDA que se puede consultar en el código fuente del programa.

IV.2.3 Programación y configuración de los dispositivos del microcontrolador

En esta sección se describe como se configuró el microcontrolador, es decir qué registros se cambiaron para activar el convertidor analógico digital, los chips selects, qué puertos eran de entrada y cuáles de salida.

IV.2.3.1 Convertidor analógico digital

Como se describe en el capitulo III, el convertidor analógico digital que proporciona el microcontrolador es de 8 bits, la referencia se fijo a 5V. Por lo tanto, podemos recibir una señal en el rango de 0V a 5V.

Para activar el convertidor analógico/digital hay que encender el bit 8 del registro OPTION del microcontrolador ubicado en la dirección $1039, y también hay que indicarle cuando queremos que haga una conversión, por lo cual hay que activar el bit 5 del registro ADCTL del microcontrolador ubicado en la dirección $1030. Esta inicialización se hace en la subrutina INI_AD y en el la subrutina LEE_Y0 respectivamente, las instrucciones para configurar se muestran a continuación:

```
LDX #$1000
BSET OPTION,X,$80
LDAA #$00
STAA ADCTL,X
```

Para el valor convertido se tiene que leer el registro ADR1 del microcontrolador ubicado en la dirección $1031, pero como estamos leyendo, un valor dentro del rango de 0 a 255, por lo cual se

debe convertir a punto flotante en el rango de 0V a 5V , entonces una vez convertido este valor a punto flotante se multiplica por la constante 5/255 y con esto tendremos el valor deseado. Todo este proceso se realiza en la subrutina llamada LEE_Y0.

IV.2.3.2 Configuración para modo expandido

La configuración en modo expandido es el modo de funcionamiento del microcontrolador con memoria externa que puede ser hasta de 64 Kbytes, por lo cual es necesario configurar los chip selects del microcontrolador.

Para activar el chip select CSI01, que es el que se usará para habilitar el *latch* que está conectado al display hay que prender los bits 7 y 8 del registro CSCTL ubicado en la dirección $105D. El bit 7 sirve para fijar la polaridad del chip select en este caso será alta, es decir, se activará con alto, el bit 8 sirve para habilitar dicho chip select, estas operaciones deben hacerse al principio del programa.

El chip select usado para habilitar la memoria RAM externa del sistema es el CSGEN (General-Purpose Chip-Select) , por lo cual hay que indicarle al microcontrolador de qué tamaño es la memoria que direccionará y a partir de qué dirección lo hará. Por lo tanto, hay que configurar el registro CSCTL (Chip-Select Control) con $00 para indicarle que esta memoria empezará a partir de $0000, y le registro CSGADR (General-Purpose Chip-Select Address Register) se le pondrá $01, con lo cual indicamos que el tamaño de la memoria será de 32 Kbytes.

Otro registro que hay que configurar es le CONFIG, donde se le indica dónde empezará la dirección de la EEPROM, y si esta estará habilitada en modo expandido, en nuestro caso esta memoria estará activa por lo tanto, a este registro se le pone $FF.

El puerto A, que es el puerto que se conecta al convertidor analógico digital, hay que configurarlo como puerto de salida, por lo tanto, es necesario escribir $FF en el registro DDRA

ubicado en la dirección $1000.

EL puerto G, que es el puerto donde se conectan las líneas de control para el display, debe ser configurado también como puerto de salida, por lo cual hay que escribir $FF en el registro DDRG ubicado en la dirección $1003.

Por último un punto importante, al principio del programa, es inicializar el stack, el cual se está inicializando en la última dirección de la RAM externa en la dirección $7FFF.

IV.2.4 PROGRAMACION DE LOS ALGORITMOS DE CONTROL

En este punto se describirá cómo se programaron los algoritmos de control que usa el dispositivo en cuestión, y que fueron mencionados en el capítulo dos de este trabajo.

Los algoritmos programados en este controlador como ya se habían mencionados son:

1.- Algoritmo PID (Proporcional, Integral, Derivativo)

2.- Algoritmo de asignación de polos y ceros

Además de estos algoritmos se incluye otra opción para la operación manual del dispositivo, en la cual el usuario puede fijar manualmente la señal de salida, es decir la señal de control .

IV.2.4.1 Algoritmo PID (Proporcional, Integral, Derivativo)

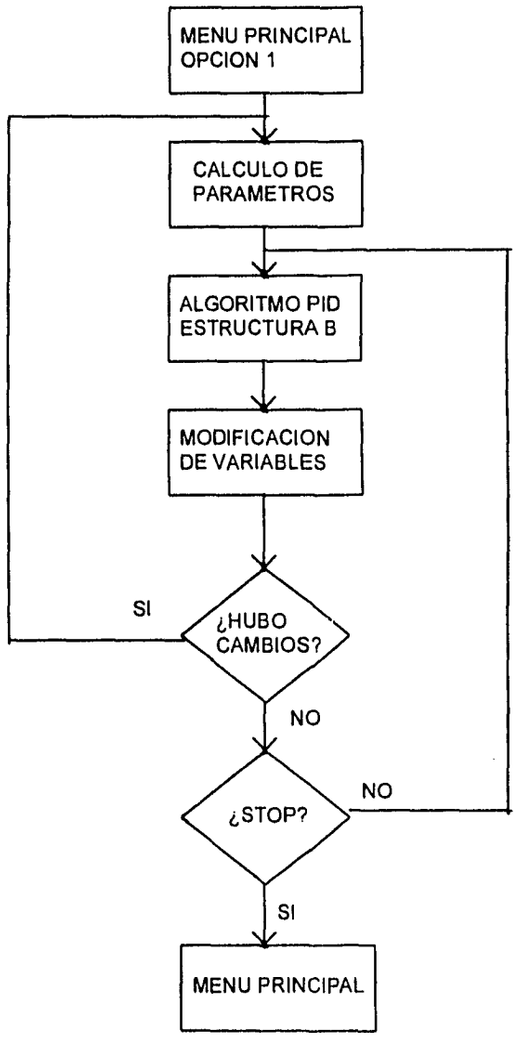
Para la implementación de este algoritmo se hizo uso de dos subrutinas principales, y las de inicialización, lectura de datos de entrada y generación de la salida.

La primer subrutina principal consiste en el cálculo de todos los coeficientes que utiliza el algoritmo, y los cuales se deben calcular cada vez que se modifica alguna de las variables del controlador como lo son: (K, Ti, Td, R). Los coeficientes que son calculados en esta subrutina para el uso del algoritmo son: (Alfa, Beta, Gama, s0, s1, s2, t0, t1, t2, q0, q1, q2), todas estas variables son calculadas mediante rutinas de punto flotante, ya que todo se esta manejando de esta manera.

La subrutina en la cual se calculan todos estos valores previos del algoritmo es llamada en el programa PID.

La segunda subrutina para llevar a cabo el funcionamiento correcto del algoritmo PID es la subrutina ESTB_PID, que es la rutina donde se calcula el valor de la señal de control tomando en cuenta dos valores anteriores de las variables Y, R y U como lo indica el algoritmo.

DIAGRAMA DE FLUJO ALGORITMO PID



Esta subrutina es una subrutina que siempre estará funcionando, una vez que se haya elegido, y sólo dejará de operar cuando el usuario elija la opción de STOP como se ha mencionado anteriormente, obviamente esta subrutina también hace uso de operaciones de punto flotante, por lo cual constantemente llama a subrutinas que realizan estas.

El algoritmo para calcular la salida del controlador, es el correspondiente a la estructura B mencionado en el capítulo dos.

En esta subrutina se llama a la subrutina que lee la entrada del convertidor analógico digital y hace la correspondiente adaptación y conversión a un valor en punto flotante para que sea compatible con las variables utilizadas en el algoritmo. También en esta subrutina se llama a la subrutina que genera la salida la cual efectúa el cálculo correspondiente para convertir el número calculado que esta en notación flotante, en un valor binario de ocho bits.

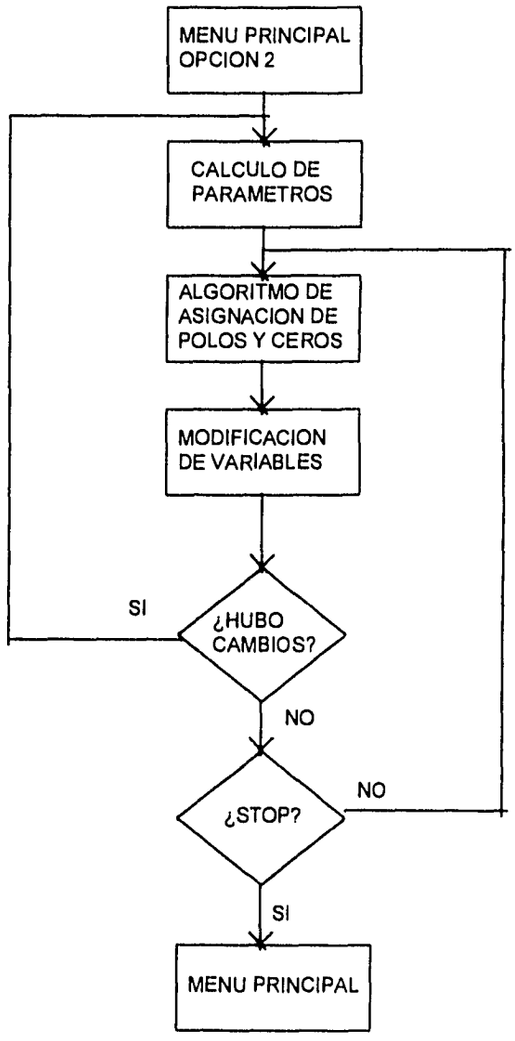
En esta parte del programa también se incluye la subrutina que detecta si se oprime alguna tecla, y de ser el caso incrementar o decrementar la variable elegida, o bien salir del algoritmo, en esta subrutina también se lleva le control de los valores máximos y mínimos de cada variable, para que estos no tomen valores fuera de la realidad o que siempre provoquen que el sistema controlado sea inestable.

Otra de las subrutina que es llamada en esta parte del programa, se encarga de actualizar las variables del sistema, es decir, almacenar los valores calculados o leídos U, Y ó R de tal manera que siempre se cuenta con los dos últimos, lo cual es un requerimiento para el correcto funcionamiento del algoritmo PID.

IV.2.4.2 Algoritmo de asignación de polos y ceros

Para la implementación de este algoritmo se hizo uso de dos subrutinas principales, y las de inicialización, lectura de datos de entrada y generación de la salida.

DIAGRAMA DE FLUJO ALGORITMO ASIGNACION DE POLOS Y CEROS



La primer subrutina principal consiste en el cálculo de todos los coeficientes que utiliza el algoritmo, y los cuales se deben calcular cada vez que se modifica alguna de las variables del controlador como lo son: (K, Tr, Mp, R, polo A, polo C, cero B). Los coeficientes que son calculados en esta subrutina para el uso del algoritmo son: (Wn, R1, P1, P2, T0, q0, q1, s0, s1), todas estas variables son calculadas mediante rutinas de punto flotante, ya que todo se está manejando de esta manera.

La subrutina en la cual se calculan todos estos valores previos del algoritmo es llamada en el programa P0YCE.

La segunda subrutina para llevar a cabo el funcionamiento correcto del algoritmo POYCE es la subrutina CICLOPYC, que es la rutina donde se calcula el valor de la señal de control tomando en cuenta un valor anterior de las variables Y, R y U como lo indica el algoritmo.

Esta subrutina es una subrutina que siempre estará funcionando, una vez que se haya elegido, y solo dejará de operar cuando el usuario elija la opción de STOP como se ha mencionado anteriormente, obviamente esta subrutina también hace uso de operaciones de punto flotante, por lo cual constantemente llama a subrutinas que realizan estas.

El algoritmo para calcular la salida del controlador, es el mencionado en el capítulo dos.

En esta parte del código se llama a la subrutina que lee la entrada del convertidor analógico digital y hace la correspondiente adaptación y conversión a un valor en punto flotante para que sea compatible con las variables utilizadas en el algoritmo. También en esta parte se llama a la subrutina que genera la salida la cual efectúa el cálculo correspondiente para convertir el número calculado que es esta en notación flotante, en un valor binario de ocho bits.

En esta parte del programa también se incluye la subrutina que detecta si se oprime alguna tecla, y de ser el caso incrementar o decrementar la variable elegida, o bien salir del algoritmo, en esta subrutina también se lleva le control de los valores máximos y mínimos de cada variable, para

que estos no tomen valores fuera de la realidad o que siempre provoquen que el sistema controlado sea inestable.

Otra de las subrutina que es llamada es esta parte del programa, se encarga de actualizar las variables del sistema, es decir, almacenar los valores calculados o leídos U, Y ó R de tal manera que siempre se cuenta con el último, lo cual es un requerimiento para el correcto funcionamiento del algoritmo de asignación de polos y ceros.

Como se puede observar para los dos algoritmos utilizados, PID y polos y ceros, se siguió el mismo estilo y estructura de programación y este con el fin de hacerlos mas entendibles y para que estos puedan compartir subrutinas de uso de común, y también para que las modificaciones u optimizaciones posteriores sean fáciles de hacer.

IV.2.4.3 Operación Manual

Como todo controlador digital comercial, este controlador también cuenta con la opción de operarlo manualmente, es decir, el usuario podrá de manera manual, mediante las teclas fijar la señal de control, y de esta manera fijar la adecuada para sus necesidades.

La estructura del programa es similar a las dos anteriores, pero en este caso no se cuenta con ningún algoritmo, por lo tanto, se implementa sólo mediante una subrutina principal que es cíclica, y se encarga de detectar si se oprimió alguna tecla y de ser el caso, incrementar o decrementar alguna variable.

Como en las otras subrutinas vistas, la manera de terminar con la operación manual del dispositivo es eligiendo la opción de STOP.

La subrutina es llamada en el programa como MANUAL y a su vez tiene dos subrutinas que utiliza para su funcionamiento que son VARIABLEMANUAL y CONVIERTEMANUAL.

V. PRUEBAS Y CONCLUSIONES

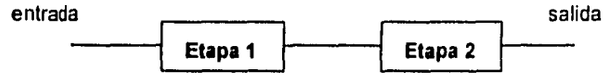
V.- PRUEBAS Y CONCLUSIONES

V.1 Pruebas

Para probar el correcto funcionamiento de los algoritmos de control implementados en el sistema digital en cuestión, se utilizó el simulador de proceso módulo G26/EV de Elettrocnica Veneta.

Con este simulador de procesos podemos obtener el comportamiento de una planta analógica, la cual será el objetivo a controlar.

Se simuló una planta de segundo orden definida por dos etapas de primer orden.



En este simulador se puede trabajar con dos tiempos de respuesta, de 10ms y 1s para una etapa y 7ms y 0.7s para la otra etapa, también se puede elegir el tipo de respuesta de las etapas, exponencial o lineal, en nuestra pruebas se usó el tipo de respuesta exponencial.

La primer etapa está definida por las siguientes ecuaciones en el dominio de Laplace:

$$W(s) = \frac{100}{S + 100} \qquad \text{para 10ms}$$

$$W(s) = \frac{1}{S + 1} \qquad \text{para 1 s}$$

La segunda etapa está definida por las siguientes ecuaciones en el dominio de Laplace:

$$W(s) = \frac{140}{S + 140} \qquad \text{para 7ms}$$

$$W(s) = \frac{1.4}{S + 1.4} \qquad \text{para 0.7s}$$

Con esta información se obtuvieron las respectivas funciones de transferencia para dicho proceso, para una respuesta rápida combinando la etapa uno de 10ms con la etapa dos de 7ms, y para una respuesta lenta mediante la unión de la etapa uno de 1s y la etapa dos con 0.7s:

Respuesta rápida:

$$H(s) = \frac{14000}{S^2 + 240\,S + 14000}$$

Respuesta lenta:

$$H(s) = \frac{1.4}{S^2 + 2.4\,S + 1.4}$$

V.1.1 Prueba del algoritmo PID

Antes de poner en funcionamiento los dispositivos y asignar valores adecuados a los parámetros K, Td y Ti, se simuló en computadora mediante la ayuda del programa SIMNON, el proceso analógico, el controlador digital y la conexión entre ellos.

Para el proceso analógico se construyó el siguiente archivo para una respuesta rápida:

```
CONTINUOUS SYSTEM LAGFAST
"Planta o proceso con respuesta exponencial y velocidad operativa FAST"
TIME t
INPUT u
OUTPUT y
STATE x1 x2
DER dx1 dx2
"Ecuaciones de estado"
dx1=x2
dx2=-a2*x1 - a1*x2 + u
"Ecuaciones de salida"
y= a2*x1
"Asignación de parámetros"
a1: 240
a2: 14000
"Asignación del valor de una entrada constante"
"u:1"
"Inicialización de variables de estado"
x1:0
x2:0
END
```

Para el proceso analógico se construyó el siguiente archivo para una respuesta lenta:

```
CONTINUOUS SYSTEM LAGSLOW
"Planta o proceso con respuesta exponencial y velocidad operativa FAST"
TIME t
INPUT u
OUTPUT y
STATE x1 x2
```

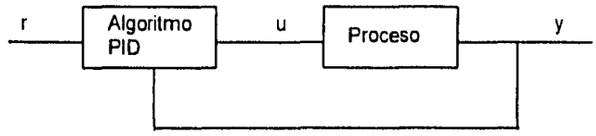
```
DER dx1 dx2
"Ecuaciones de estado"
dx1=x2
dx2=-a2*x1 - a1*x2 + u

"Ecuaciones de salida"
y= a2*x1
"Asignación de parámetros"
a1: 2.4
a2: 1.4
"Asignación del valor de una entrada constante"
"u:1"
"Inicialización de variables de estado"
x1:0
x2:0
END
```

Para el controlador digital considerando el algoritmo PID, se construyó el siguiente archivo:

```
DISCRETE SYSTEM CTRLPID
TIME t
TSAMP ts
INPUT y
OUTPUT u
STATE u1 u2 r1 r2 y1 y2
NEW nu1 nu2 nr1 nr2 ny1 ny2
"Ecuaciones de estado"
nu1 = u
nu2 = u1
nr1 = r
nr2 = r1
ny1 = y
ny2 = y1
" Operaciones "
alfa   = K*h/Ti
unobeta = K*Td/h
gamma  = Td/(N*h) + 1
s0     = gamma
s1     = (1 - 2*gamma)
s2     = (gamma -1)
q0     = K*gamma + unobeta
q1     = K*s1 + alfa*gamma - 2*unobeta
q2     = (K - alfa)*(gamma - 1) + unobeta
t0     = K*gamma
t1     = K*s1 + alfa*gamma
t2     = (K - alfa)*(gamma - 1)
"Ecuación de salida"
salida = (-s2*u2 -s1*u1 +t0*r +t1*r1 +t2*r2 -q2*y2 -q1*y1 -q0*y)/s0
u = salida
"Incremento de tiempo"
ts = t + h
N  : 10          "Asignación de parámetros"
K  : 0.4
Ti : 0.02
```

```
Td  : 0.02
r   : 1
h   : 0.05
"y  : 1"
u1  : 1          "Inicialización de variables de estado"
u2  : 1
r1  : 1
r2  : 1
y1  : 1
y2  : 1
end
```

Se hizo la conexión entre el proceso y el controlador digital de acuerdo al siguiente diagrama:

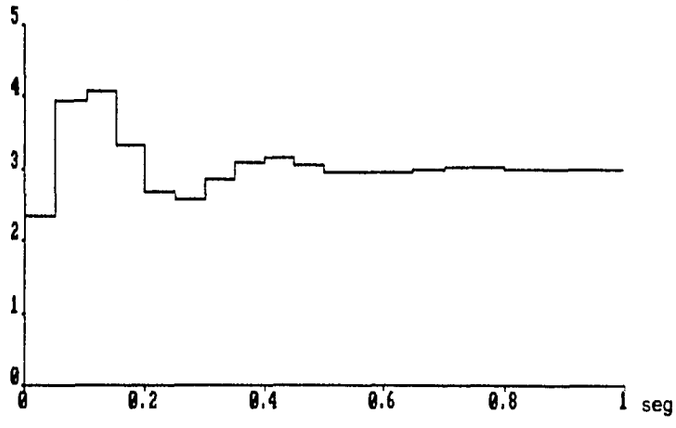


donde "r" es la referencia, "u" es la señal de control y "y" es la salida. El archivo es el siguiente:

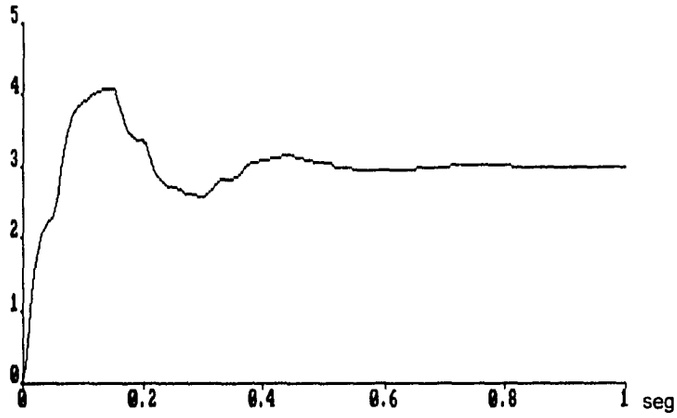
```
CONNECTING SYSTEM TESTPID
TIME t
u[LAGFAST] = u[CTRLPID]
y[CTRLPID] = y[LAGFAST]
END
```

De las simulaciones realizadas para el proceso con respuesta rápida observamos que los valores adecuados para los parámetros son K = 0.4, Ti = 0.02s, Td = 0.02s y una referencia igual a 3 como se muestra en las siguientes gráficas:

Señal de control:(Volts)

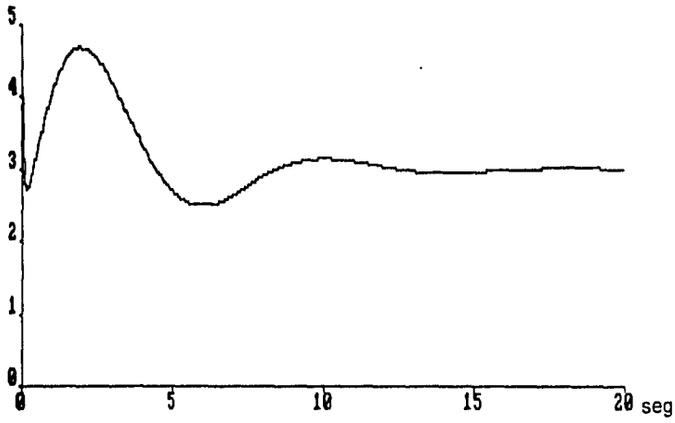


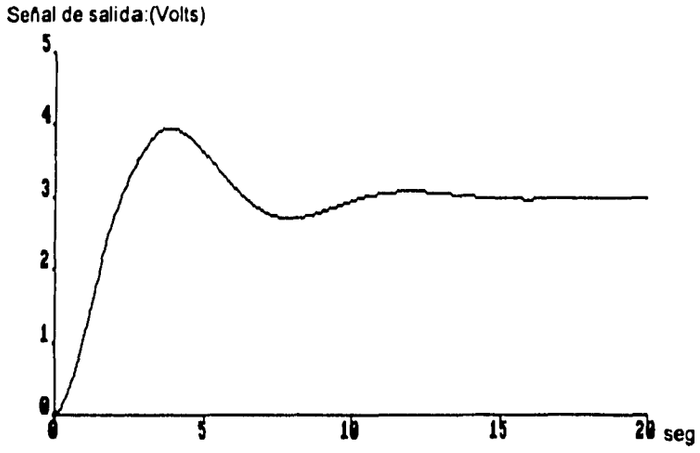
Señal de salida:(Volts)



De acuerdo a las simulaciones realizadas para el proceso con respuesta lenta observamos que los valores adecuados para los parámetros son K = 0.4, Ti = 0.4s, Td = 0.4s y una referencia igual a 3 como se muestra en las siguientes gráficas:

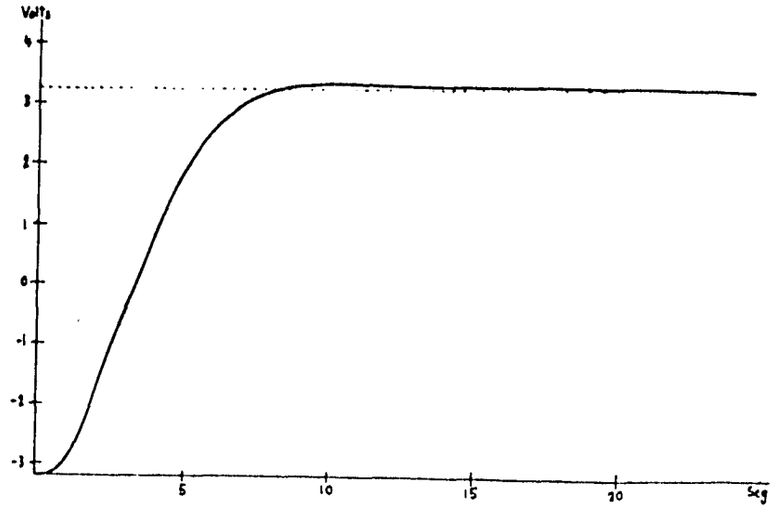
Señal de control:(Volts)



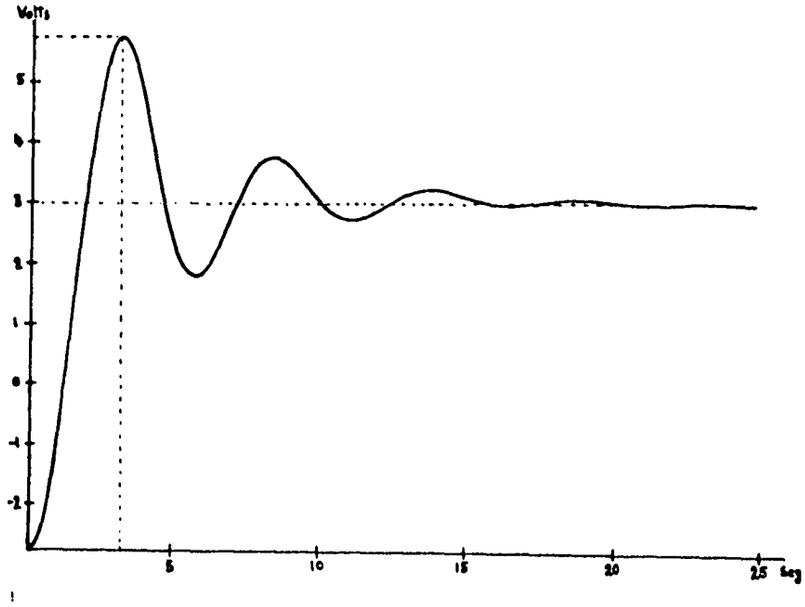


V.1.2 Prueba del dispositivo con el algoritmo PID

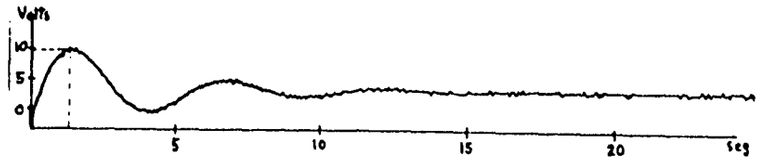
Para probar el controlador digital se utilizó el simulador G26 y un graficador en donde se dibujó la señal de control y la respuesta de la planta para las diferentes pruebas realizadas. En el simulador se utilizó la configuración descrita al principio de este capítulo, en el controlador digital se eligió el algoritmo de control PID. Se empezaron a hacer las pruebas con los siguientes parámetros para el algoritmo PID: Referencia = 3 Volts, Tiempo de Integración (Ti)= 1 seg, Tiempo de Derivación (Td) = 0.4 seg y Ganancia (K)= 0.4 .Con estos valores se obtuvo la siguiente respuesta del simulador G26 para un proceso de respuesta lenta (Señal de salida):



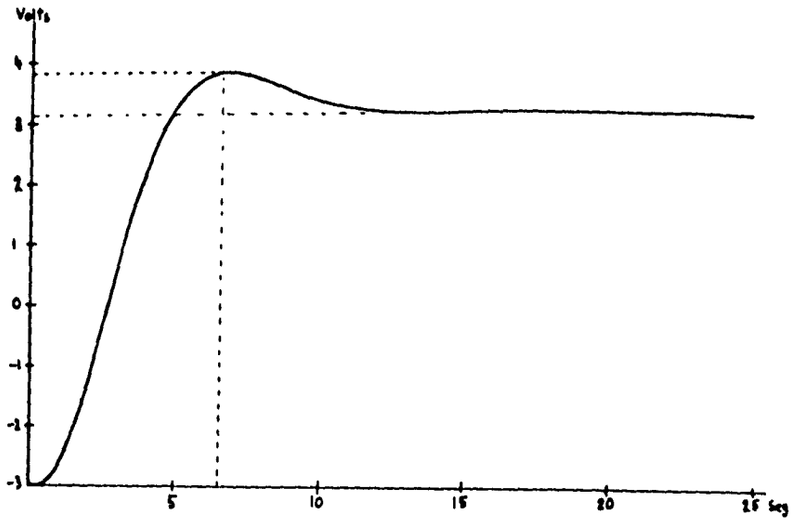
Cambiando la ganancia a K=1 se obtiene la siguiente señal de salida:



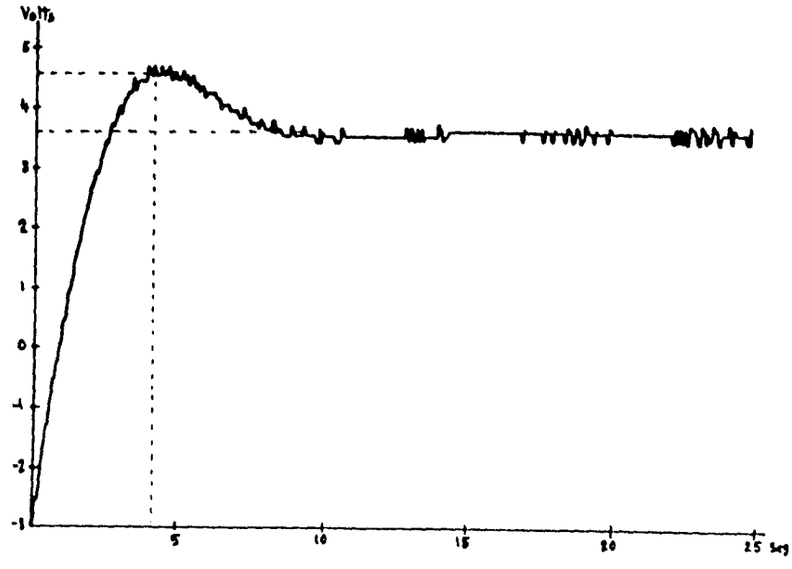
y la señal de control tiene el siguiente comportamiento:



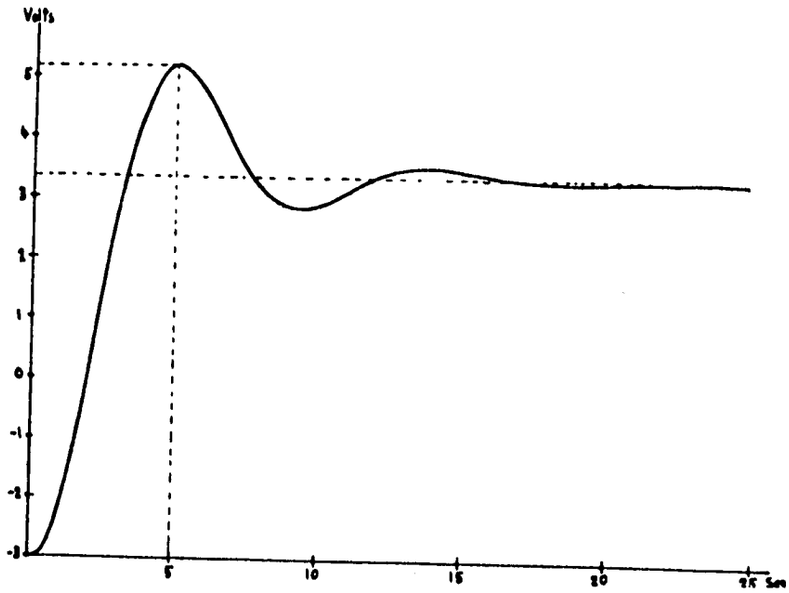
Regresando K = 0.4 y haciendo Ti = 0.7 seg, se obtiene la siguiente señal de salida:



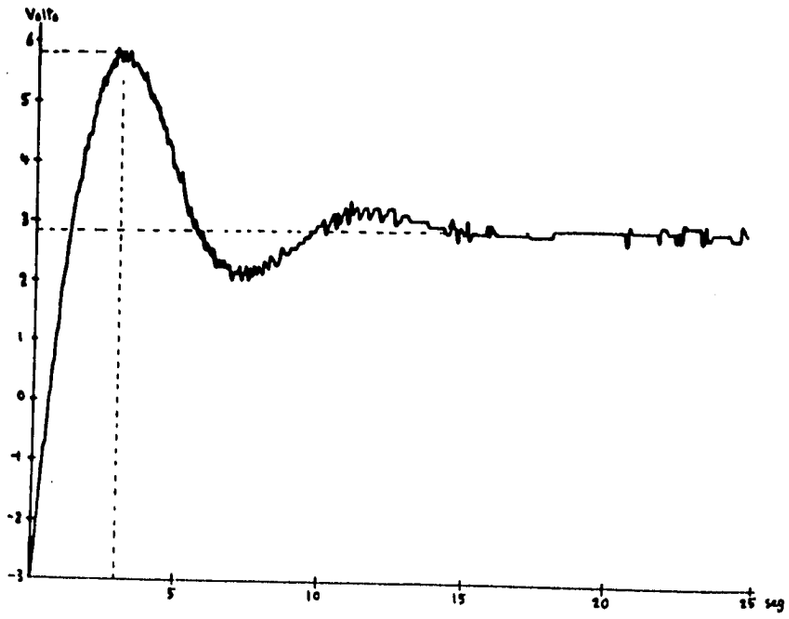
y la señal de control tiene el siguiente comportamiento:



Ahora cambiando Ti = 0.4 seg y Td = 0.8 seg se obtiene la siguiente salida:



y la señal de control tiene el siguiente comportamiento:



De acuerdo a las gráficas obtenidas anteriormente podemos observar que los valores óptimos para llevar a cabo un control adecuado obteniendo poco sobrepaso y un tiempo de respuesta rápido, es decir, un corto tiempo de estado transitorio, son con los que se iniciaron las pruebas Ti = 1seg, Td= 0.4 seg y K = 0.4 para una referencia de 3 Volts.

Cuando varió el valor de K notamos que el sobrepaso aumento considerablemente, también aumento el tiempo en el estado transitorio, así como la frecuencia en las oscilaciones, también podemos notar que la señal de control alcanzó un valor más alto al inicio del estado transitorio.

Cuando decrementamos el valor de Ti, aumento un poco el sobrepaso, la señal de control no se incremento demasiado, por lo que con estos parámetros también se obtuvo una respuesta apropiada.

Por último cuando se incremento el valor de Td observamos que aumento el sobrepaso, el tiempo en estado transitorio, así como el pico de la señal de control en estado transitorio.

Con las pruebas realizadas podemos concluir que el comportamiento del controlador digital es adecuado ya que al variar los parámetros del algoritmo PID se obtienen las respuestas esperadas, porque como pudimos observar al variar la constante del control proporcional (K) obtenemos una respuesta más rápida del proceso, es decir, se alcanza más rápidamente por primera vez el valor de la referencia sin embargo tenemos un sobrepaso mayor, así como oscilaciones lo que provoca que aumente el tiempo en estado transitorio. Por otro lado cuando disminuimos el Ti, también logramos alcanzar más rápidamente el valor de la referencia por primera vez pero provocando mayor sobrepaso. Finalmente cuando se cambió el valor de Td se obtuvo la misma respuesta, ya que debido a que el control derivativo nos trata de proporcionar la estabilidad del sistema el aumentar el tiempo de derivación obtenemos más oscilaciones.

En lo que se refiere a la señal de control observamos pequeñas oscilaciones esto se debe a las aproximaciones de valores de punto flotante a un número entero entre 0 y 255, así como la aproximación del convertidor digital-analógico, y la amplificación de esta señal para que quede en el rango de -10 Volts a 10 Volts. Sin embargo este pequeño ruido no afecta al control que se esta llevando a cabo, de esta manera podemos obtener el control deseado.

V.1.3 Prueba del algoritmo de asignación de polos y ceros

Debido a los requerimientos de este algoritmo, se tienen que calcular los valores de los parámetros K, a, b y c, que van en función al proceso analógico a controlar.

Con ayuda del método de aproximación "Reten de Orden Cero" (ROC) obtuvimos el equivalente de la función de transferencia del proceso analógico, en el dominio discreto, de acuerdo a las siguientes fórmulas:

Teniendo:

$$\frac{ab}{(s-a)(s-b)}$$

Su equivalente en el dominio discreto es:

$$\frac{b_1 Z + b_2}{z^2 + a_1 z + a_2}$$

donde:

$$b_1 = \frac{b(1-e^{-ah}) - a(1-e^{-bh})}{b-a}$$

$$b_2 = \frac{a(1-e^{-bh})e^{-ah} - b(1-e^{-ah})e^{-bh}}{b-a}$$

$$a_1 = -(e^{-ah} + e^{-bh})$$

$$a_2 = e^{-(a+b)h}$$

donde h es el tiempo de muestreo.

Para el proceso de respuesta rápida obtuvimos:

$$Gp(z) = \frac{0.97869(z + 0.01395)}{(z - 0.00673)(z - 0.00091)}$$

Para el proceso de respuesta lenta obtuvimos:

$$Gp(z) = \frac{0.00168(z + 0.96078)}{(z - 0.95122)(z - 0.93293)}$$

Para asignar valores adecuados de los parámetros Mp y Tr, nuevamente nos auxiliamos de la simulación en computadora, esto se hizo utilizando los mismos archivos para los procesos analógicos y construyendo el siguiente archivo para el controlador digital:

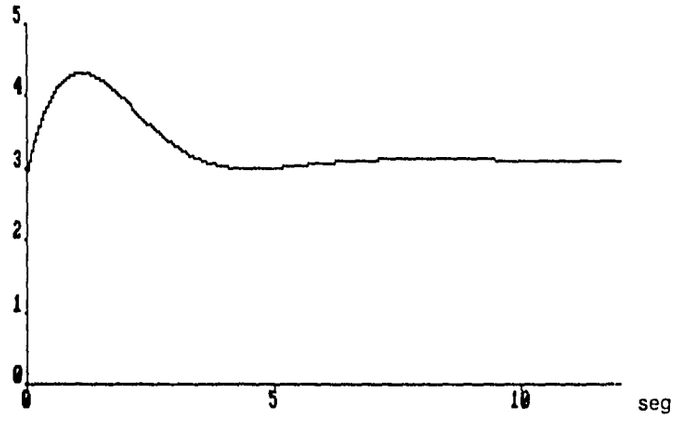
```
DISCRETE SYSTEM CTRLPYC
TIME t
TSAMP ts
INPUT y
OUTPUT u
STATE y1 u1
NEW ny1 nu1

SI = -LN(MP)/SQRT(3.1416*3.1416 + LN(MP)*LN(MP))
W  = (3.1416 - ARCCOS(SI))/TR
WN = W / SQRT(1-SI*SI)
P1 = -2*EXP(-SI*WN*dt)*COS(W*DT)
P2 = EXP(-SI*WN*dt)*EXP(-SI*WN*dt)
F1 = (C-B)
F2 = (A-B)
F3 = (A*A*A + P1*A*A + P2*A)
F4 = (C*C*C + P1*C*C + P2*C)
F5 = K*(C-A)*F1*F2
F6 = F2*F4
F7 = F1*F3
Q0 = (F6-F7)/F5
Q1 = (C*F7-A*F6)/F5
T0 = (1+P1+P2)/(K*(C-B))
R1 = -B + B*(B*B + P1*B + P2)/((B-C)*(B-A))

ny1 = y
nu1 = u

u = (T0*r -Q0*y -Q1*y1 -R1*u1)

ts = t + dt

y1:0
u1:0

dt:0.05
a:0.9512295
c:0.9323939
MP:0.1
TR:1.4
b:-0.9607891
K:0.001681564
r:1
END
```

Así como el archivo de conexión:

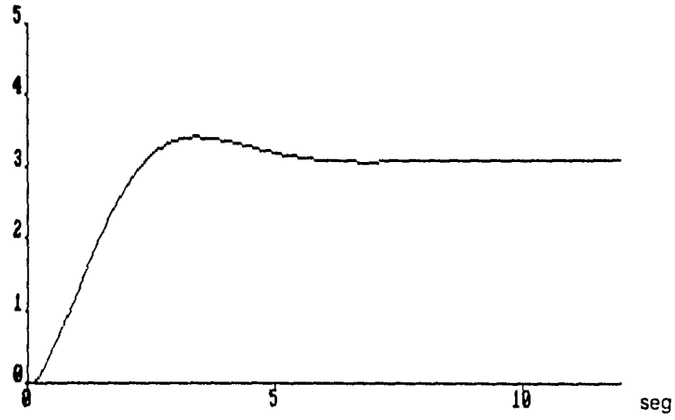
```
CONNECTING SYSTEM TESTPYC
TIME t
u[LAGFAST] = u[CTRLPID]
y[CTRLPID] = y[LAGFAST]
END
```

De acuerdo a las simulaciones realizadas para el proceso con respuesta lenta observamos que los valores adecuados para los parámetros son Mp = 0.1, Tr = 2.4s y una referencia igual a 3 como se muestra en las siguientes gráficas:

Señal de control:(Volts)

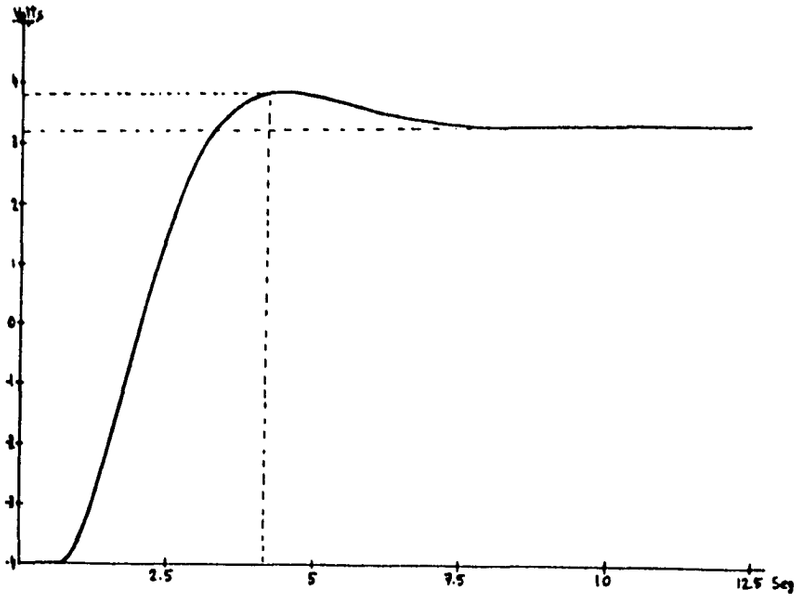


Señal de salida:(Volts)

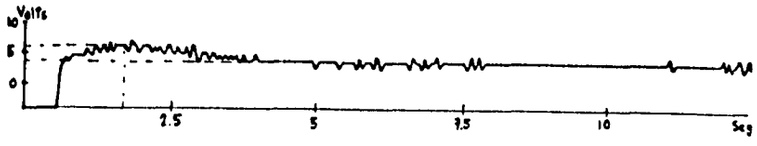


V.1.4 Prueba del dispositivo con el algoritmo de asignación de polos y ceros

Para probar el controlador digital pero ahora con el algoritmo de asignación de polos y ceros en el simulador se utilizó la configuración descrita al principio de este capítulo para respuesta lenta, en el controlador digital se eligió el algoritmo de control de asignación de polos y ceros. Se hizo la prueba con los siguientes parámetros referencia = 3 Volts, Mp = 10% y Tr = 2.4 seg, obteniéndose en el graficador la siguiente señal de salida:



y el comportamiento de la señal de control fue el siguiente:



De acuerdo a las gráficas obtenidas podemos observar que los parámetros que fijamos en nuestro controlador (Mp y Tr) son bastante aproximados a los obtenidos en estas gráficas.

En lo que se refiere a la señal de control igual que en algoritmo PID observamos pequeñas oscilaciones esto se debe también a las aproximaciones de valores de punto flotante a un número entero entre 0 y 255, así como la aproximación del convertidor digital-analógico, y la amplificación de esta señal para que quede en el rango de -10 Volts a 10 Volts. Sin embargo este pequeño ruido no afecta al control que se esta llevando a cabo, de esta manera podemos obtener el control deseado.

En todas las simulaciones se usó una frecuencia de muestreo de 20 Hz que es la frecuencia de operación del controlador digital implementado.

V.2 Conclusiones

Como se puede observar con ambos algoritmos logramos un control aceptable del proceso de respuesta lenta, ya que para el proceso de respuesta rápida sólo se logró efectuar con el algoritmo PID, por que en el caso del algoritmo de asignación de polos y ceros no fue posible debido a que en el equivalente en el dominio discreto del proceso analógico con un tiempo de muestreo de 0.05 segundos se obtienen valores muy pequeños de los polos y del cero, los cuales son difíciles de manejar por la resolución de los cálculos.

Por otra parte, también se puede concluir que para el caso del proceso de respuesta lenta, con el algoritmo de asignación de polos y ceros, se logra obtener más rápido el valor de la referencia y la señal de control es más constante que en el caso del algoritmo PID, haciendo notar que la señal de control en el algoritmo de asignación de polos y ceros depende del tiempo de levantamiento (Tr) y del sobrepaso (Mp) que asignemos, ya que entre más pequeño sea el tiempo de levantamiento tendremos, en algunos momentos del estado transitorio, una señal de control mas grande. En el caso del algoritmo PID si queremos obtener más rápido el valor de la referencia cambiando los parámetros Ti, Td y K, obtenemos una señal de control con mayor amplitud, por lo que no es conveniente hacer esto.

Se recomienda usar el algoritmo PID cuando tengamos pocos datos del comportamiento de algún proceso debido a que es más rápido establecer el valor de las variables K, Td y Ti para obtener el control deseado.

En el caso del algoritmo de asignación de polos y ceros logre controlar algún proceso necesitamos más datos sobre el comportamiento de algún proceso como lo son los ceros y polos más dominantes para poder establecer su función de transferencia, no por esto hace menos eficiente y confiable este algoritmo.

Es aconsejable que antes de poner en funcionamiento el controlador digital implementado, se usen las herramientas de simulación en computadora con el propósito de obtener más rápidamente los parámetros que proporcionarán el control requerido y evitar estar jugando con la respuesta del proceso en manera real.

De acuerdo al diseño de este controlador y las facilidades que nos proporcionan los sistemas digitales, es posible implementar otros algoritmos de control en este dispositivo, sin tener que implementar otra arquitectura física, y sólo tener que programarlos, por ejemplo en el caso del algoritmo de asignación de polos y ceros, cuando la planta o proceso a controlar no sea igual al modelo de segundo orden manejado en el capítulo dos, se puede obtener otro algoritmo considerando el modelo correspondiente y de esta manera obtener el control deseado.

En caso de que se necesite diseñar un controlador digital con una frecuencia de muestreo y de respuesta, con valor más grande, podemos basarnos en la arquitectura diseñada en este trabajo agregando componentes físicos, como puede ser un procesador digital de señales (DSP), que pueden interactuar en conjunto con el microcontrolador MC68HC11.

Como equipo de laboratorio es muy ilustrativo ya que en este dispositivo es posible variar diferentes parámetros que intervienen en los algoritmos de control y de esta manera entender de una manera práctica qué significa cada uno de estos, además permite probar nuevos algoritmos de control.

BIBLIOGRAFÍA

-Computer Controlled Systems Theory and Design.
Karl J. Astrom, Bjorn Wittenmark. ed. Prentice Hall.

-MC68HC11 Reference Manual
Motorola

-MC68HC11 Floating-Point Package
Motorola

-MC68HC11 Technical Summary
Motorola

-AND Display Products Catalog 1991

-Manual de Laboratorio de Control Digital
Departamento de Ingeniería de Control
Facultad de Ingeniería UNAM.

-Modulo G26/EV Simulador de Proceso
ElecttronicaVeneta.

-Apuntes de Métodos Numéricos
Facultad de Ingeniería UNAM