UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

"DISEÑO DE UN SISTEMA DE CONTROL DE POSICIÓN DE UN MOTOR DC USANDO PROCESADOR DIGITAL DE SEÑALES"

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO

PRESENTADO POR:

JORGE LUIS INCA RODRÍGUEZ

PROMOCION 1987 - I

LIMA – PERU

2 002

A mis padres, Eleodoro y Alberta porque me alentaron en todo momento.

DISEÑO DE UN SISTEMA DE CONTROL DE POSICIÓN DE UN MOTOR DC USANDO PROCESADOR DIGITAL DE SEÑALES

SUMARIO

En esta tesis se hace el estudio de la configuración del controlador auto-ajustable(CAA) representado en la figura 4.1, este sistema de control de posición en que se presenta la implementación de la simulación utilizando un nuevo dispositivo, el procesador digital de señales (DSP), que se encuentra en la fase inicial de sus implementaciones en el laboratorio, que trabaja como microcontrolador con aritmética de punto fijo y que responde con mayor exactitud en régimen fraccionario.

Debido a su alta velocidad en el procesamiento digital de información y al procesamiento digital de señal en tiempo real, lo hace recomendable para elaborar algoritmos simplificados, dichos algoritmos fueron extraídos de programas en lenguaje C++ utilizando una microcomputadora Pentium, que ya probados, pudieron ser reproducidos en lenguaje ensamblador del DSP, estos algoritmos como el de estimación de parámetros, el modelo de representación de proceso y una ley de control, así como la técnica de mínimos cuadrados, permitieron con la ayuda de software de Matlab para el cálculo de constantes optimizadas, llegar a la simulación del programa de control de posición en lenguaje ensamblador del DSP, logrando así utilizar la comunicación con los puertos paralelos, el PLL (oscilador por enganche de fase) y las instrucciones de simplificación de cálculos del DSP.

ÍNDICE

PRÓLOGO

La presente tesis que consta de seis capítulos y tres anexos, desarrolla el control digital de posición de un sistema con motor de corriente continua sujeto a una carga no lineal, que emplea control adaptivo y auto-ajustable aplicado al sistema, usando un nuevo procesador, el Procesador Digital de Señales, más conocido por sus siglas como DSP.

En el capítulo I se formula el problema de control, además se tiene los símbolos, variables y parámetros utilizados para resolver el sistema de con - trol de posición. En el capítulo II se presenta la descripción de la estructura y modelo de la planta constituida por un motor de corriente continua de engra - najes, que es no-lineal teniendo en cuenta la carga, para proceder al mode - lamiento de la planta mediante planteamiento de ecuaciones que manejadas adecuadamente obtenemos el modelo matemático de la planta, también se detallan los elementos de comunicación de la planta, como son el generador PWM, codificador óptico y PLD sensores y tarjeta de transmisión de datos, además se hace el alcance del procesador digital de señales utilizado y de su interface implementada. En el capítulo III se tiene la linealización y dis - cretización del modelo matemático obtenido en el capítulo anterior, se hace uso de software de Matlab y de su interface gráfica Simulink para simular el sistema, obteniendo su respuesta y analizando la estabilidad, además se

hace énfasis en el uso de la frecuencia de operación del procesador digital de señales utilizado y del registro de PLL, para variar la frecuencia de operación. En el capítulo IV se hace el estudio teórico utilizado en el controlador y se tiene los diagramas de flujo de los algoritmos del programa. En el capítulo V se presenta la implementación utilizada del hardware para obtener los datos del modelo. En el capítulo VI se desarrolla el software y se presentan los resultados. Luego se tienen las conclusiones. En el Anexo A se presenta el método de mínimos cuadrados, en el Anexo B se obtiene Jeff y beff . Finalmente en el Anexo C se muestra el listado de los programas realizados.

Deseo expresar mi agradecimiento, al Ingeniero Luis Figueroa Santos por su paciente asesoría, y al Ingeniero Agustín Gutiérrez Paucar por sus in - dicaciones de la parte eléctrica del motor de corriente continua sin cuya ayu - da no hubiera sido posible el conocimiento adecuado y preciso del subsiste - ma eléctrico para manejar la simulación del modelo de la planta en Simulink. Asimismo agradecer a la Escuela de posgrado de la Facultad de Ingeniería Eléctrica y Electrónica, por haberme permitido el uso de su biblioteca espe - cializada, así como también del ambiente de la sala de proyectos para poder probar el sistema de control.

CAPÍTULO I
FORMULACIÓN DEL PROBLEMA

1.1 Planteamiento del problema del control de posición

El problema es el siguiente utilizar la tarjeta de microcontrolador DSP56002EVM para implementar el sistema de Control adaptivo auto-ajus .- table discreto para un motor de corriente continua sujeto a una carga no lineal que es una varilla acoplada a su eje, aplicado a un control de posición angular y trabajado en radianes. El sistema debe reducir al mínimo la dife - rencia entre la salida y la señal de referencia como en la referencia [12]. La entrada al sistema es el voltaje de armadura del motor y la salida es la posición angular de la varilla. El control adaptivo con modelo de referencia paralelo es aquel situado en paralelo al sistema de bucle cerrado tal cual la referencia [7] que en principio puede resolver nuestro problema de control de seguimiento durante el primer segundo, que es el tiempo prudente obtenido por experimentación para que el sistema llegue a la posición deseada, cuando no se ha llegado aún a la posición deseada, utilizando el método de regresión lineal se impide la inestabilidad producida por sobre impulsos, que son valores de posición mayores que la posición deseada(en el primer momento se tuvo que superar el problema de un marcado sobre impulso que desestabilizó el control de posición), controlando que la posición no supere a la referencia variable en el tiempo sobre todo en los primeros instantes del

primer segundo, cuando la posición sobrepase a la referencia, varios instantes después de iniciado el suceso, la posición se hace igual a la referencia, que es el mejor valor estimado, que debería tener la posición en ese instante, la posición será igual a la referencia hasta que la referencia, que es igual a la posición, llegue a la deseada de 45°, elegida así porque es el valor más alto de ángulo notable, cuyo valor en radianes es menor que la unidad, luego de lo cual el sistema debe mantener esta posición de 45°. En los problemas de regulación, que consisten en conservar la posición desea - da, se tiene que se aceptan señales gaussianas para el control como en referencia [7]. Obtenemos señales aleatorias de control por que el error esti - mado, diferencia entre la salida del modelo paralelo de referencia y la del sistema, converge a cero durante un ciclo de discretización también en la referencia [7].

El trabajo de está tesis para obtener un sistema de control de posición para un ángulo determinado que, además pudiese ser implementado en el procesador digital de señales utilizado, dos aspectos se tomaron en cuenta que son:

1. Un estimador cuadrático, linealizado en el tiempo y discreto, utili - zando el filtro estacionario de Kalman discreto.
2. Un regulador cuadrático, linealizado en el tiempo y discreto, que minimice la función de costos y calcule la matriz de ganancia de la ley de realimentación.

Adicionalmente, el problema de sobre impulso inicial tuvo que ser re - suelto aplicando el método de regresión lineal de la referencia [13], que no

permite que la salida de posición deseada sobrepase en ningún caso a la referencia, expresando las variables de posición (y) y error (e), en función de otras como la referencia (r) y la señal de control (u), empleando mínimos cuadrados, en que pudiéndose presagiar el comportamiento de las señales se pretende que la señal de error y de la posición tengan el mismo signo, con lo que la señal del error debe tender a cero y la señal de posición a la deseada; finalmente un criterio parecido es aplicable a la señal de control, esto completa la estrategia de control.

1.2 Descripción de las variables

Las variables que se utilizaron para describir el sistema de control de posición auto-ajustable se muestran en la tabla 1.1.

Para la parte teórica en especial del capítulo IV se tienen las siguien - tes variables:

$\vec{\Psi}$ = Vector de medidas.

$\vec{\theta}$ = Vector de parámetros.

$\hat{A}$ y $\hat{B}$ = Matrices de estimación.

$\hat{x}^+$ y $\hat{x}^-$ = Estimaciones de estados.

$\Sigma(k)$ = Solución única de la ecuación matricial de Riccati.

$\Theta(k)$ y $\Xi(k)$ = Covariancias de las perturbaciones υ y ω.

1.3 Valores de los parámetros

En la tabla 1.2 se dan los valores de los parámetros del sistema que emplearemos como datos, obtenidos de hojas de especificaciones y por me - diciones directamente.

Símbolo	Descripción
b_L	Coeficiente de fricción viscosa de la carga
b_M	Coeficiente de fricción viscosa del motor
$f(.)$	Modela el efecto de la fricción
J_L	Momento de inercia de la carga
J_M	Momento de inercia del eje primario del motor
L_0	Longitud de la varilla
m	Masa de la varilla
M	Masa de la esfera
R_0	Radio de la esfera
T_M	Torque del Motor
T_L	Torque producido por la rotación de las cargas
T_E	Torque de los pesos de las cargas
T_C	Torque de fricción estática y de Coulomb
θ	Posición angular del eje reducido del motor
θ_M	Posición angular del eje principal del motor
ω	Velocidad angular del eje reducido del motor
ω_M	Velocidad angular del eje principal del motor

Tabla 1.1: Descripción de las variables.

Símbolo	Parámetro	Valor
b_L	Fricción viscosa del eje secundario del motor	No se tomó en cuenta
beff	Fricción viscosa equivalente referida a la carga	$7.05x10^{-5}$ Nm/Rads^{-1}
b_M	Fricción viscosa del eje primario del motor	$1.83x10^{-6}$ Nm/Rads^{-1}
E	Constante de voltaje contra electromotriz	$31.0352x10^{-3}$ Vs/Rad
g	Gravedad	9.8m/s^2
J_L	Inercia del eje secundario del motor	No se tomó en cuenta
Jeff	Inercia equivalente referida a la carga	$5.63x10^{-5}$ Kgm^2
J_M	Inercia del eje primario del motor	$1.9066x10^{-6}$ Kgm^2
Kac	Ganancia del driver del motor	14.9
Kt	Constante del torque del motor	$31.071x10^{-3}$ Nm/A
L	Inductancia del motor	$4.64x10^{-3}$ H
Lo	Longitud de la varilla	0.776m
m	Masa de la varilla	0.06377Kg
n	Factor de reducción de velocidad	19.741
R	Resistencia de armadura del motor	7.38 Ω

Tabla 1.2: Valores de los parámetros del sistema

CAPÍTULO II
ESTRUCTURA Y MODELO DEL SISTEMA

La implementación física del sistema necesaria para el control de posición CAA se tiene en la figura 2.1, consiste de un generador de señal PWM (modulación por ancho de pulsos), un amplificador tipo H; la planta constituida por un motor de corriente continua; el sensor PLD (dispositivo lógico programable), con un codificador óptico tipo incremental que sensa la posición y signo; una interface Lab-PC+ de entrada y salida; un PLD (dispositivo lógico programable) capaz de brindar un código adecuado para la interface Lab-PC+ de la señal sensada por el codificador óptico; y una PC compatible con microprocesador Pentium.

2.1 Modelamiento de la planta del sistema

2.1.1 Subsistema mecánico

El subsistema mecánico de la figura 2.2 puede ser modelado utilizando la segunda ley de Newton aplicada al movimiento rotacional de K partículas de momento de inercia J_j, j=1,...,K, sometidas a N torques externos. T_i, i=1,..., N, dada por la ecuación (2.1) siguiente:

$$\sum_{i=1}^{N} T_i = \sum_{j=1}^{K} J_j \dot{\omega}_j \qquad (2.1)$$

Tomando como referencia el eje reducido del motor como se mani – fiesta en la referencia [15], la posición de reposo como la del péndulo con - vencional y como positivo el sentido de giro antihorario, el torque resultante del movimiento rotacional del motor puede plantearse de la forma siguiente:

$$nT = nT_M + T_E + T_L + nT_C \tag{2.2}$$

Donde: n=a/b: Relación de vueltas en los engranajes y nT: torque total reflejado a la salida y los otros torques son los que se describen en la sección 1.2, a continuación definimos el torque del motor y la relación de velocidades dada por: $T_M = J_M \dot{\omega}_M + b_M \omega_M$ con $\omega_M = n\omega$ tendremos:

$$T_M = nJ_M \dot{\omega} + nb_M \omega \tag{2.3}$$

$$T_E = (ML_0^2 + \frac{2}{5}MR_0^2 + \frac{1}{3}mL_0^2 + J_L)\dot{\omega} + b_L \omega \tag{2.4}$$

$$T_L = gL_0(M + \frac{m}{2})\text{sen}\theta \tag{2.5}$$

$$T_c = f(n\omega) \tag{2.6}$$

$$Jeff = nJ_M + \frac{J_L}{n}$$

$$beff = nb_M + \frac{b_L}{n} \tag{2.7}$$

El torque en la reducción del eje del motor viene dado por:

$$T = [Jeff + \frac{1}{n}\left(ML_0^2 + \frac{2}{5}MR_0^2 + \frac{1}{3}mL_0^2\right)]\dot{\omega} + \omega beff + \frac{gL_0}{n}\left(M + \frac{m}{2}\right)\text{sen}\theta + f(n\omega) \tag{2.8}$$

En la ecuación (2.8) para simplificar hagamos los cambios dados por las re - laciones siguientes:

$$N = Jeff + \frac{1}{n}(ML_0^2 + \frac{2}{5}MR_0^2\frac{1}{3}mL_0^2)$$

$$A = \frac{gL_0}{n}(M + \frac{m}{2}) \tag{2.9}$$

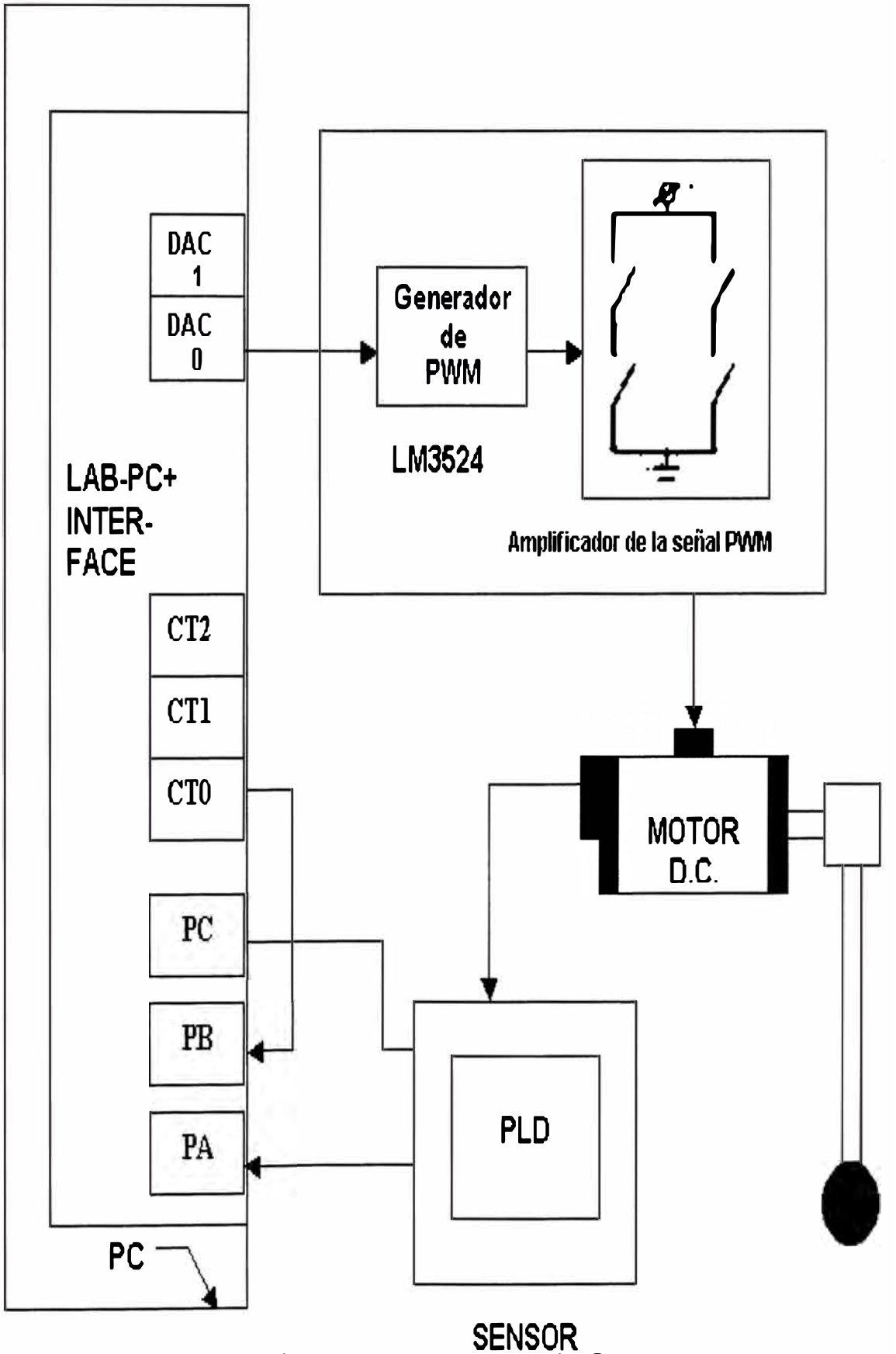

Figura 2.1: Implementación del control de posición CAA.

Reemplazando las ecuaciones (2.9) en la ecuación (2.8) obtenemos:

$$T = N\dot{\omega} + beff\,\omega + Asen\theta + f(n\omega) \qquad (2.10)$$

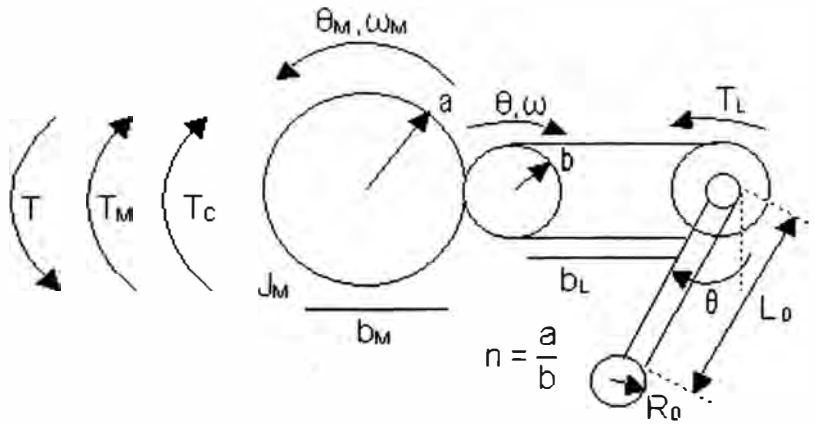

Figura 2.2: Subsistema mecánico. En la transmisión del engranaje reductor del motor, al cual va unido solidariamente la carga de la varilla la cual puede tener montado masas, todo el conjunto girando a la misma velocidad angu - lar y el mismo ángulo que el eje reducido del motor.

2.1.2 Subsistema eléctrico

En el subsistema eléctrico como se muestra en la figura 2.3, la velo - cidad del motor está controlada por la tensión de armadura, la cual se puede expresar con la siguiente ecuación:

$$Va = IR + L\dot{I} + e_b \qquad (2.11)$$

Siendo e_b la fuerza contraelectromotriz, igual al producto de la cons - tante de la fuerza contraelectromotriz (E) por la velocidad angular del prima - rio del eje del motor (ω_M). Así tenemos $e_b = E\omega_M$, reemplazando esta última ecuación en la ecuación (2.11), obtenemos:

$$Va = IR + L\dot{I} + E\omega_M \qquad (2.12)$$

Con la relación entre ω_M y ω se tiene la ecuación (2.13) siguiente:

$$Va = IR + L\dot{I} + nE\omega \qquad (2.13)$$

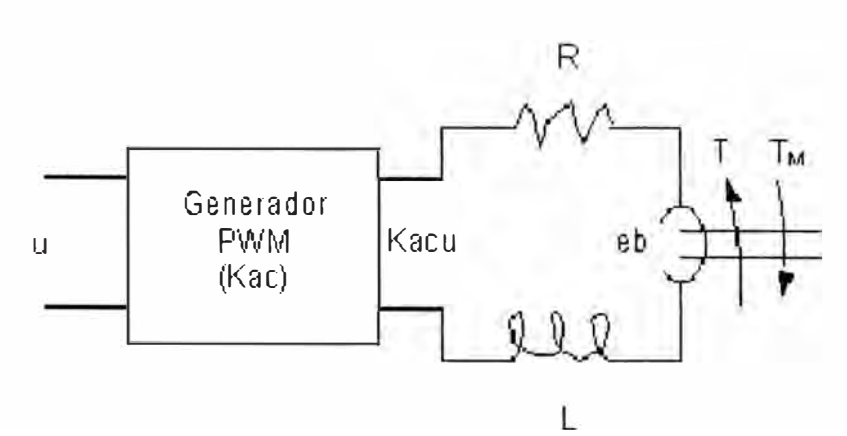

Figura 2.3: Subsistema eléctrico. Constituido por un circuito de armadura (R y L), que recibe una señal u amplificada por un generador PWM de ganancia Kac, generando la fuerza contraelectromotriz eb, que acciona el rotor produ - ciendo un torque T y una reacción del motor en un torque T_M.

En la figura 2.4 se muestra el esquema total de la planta descrita, la cual está constituida por el subsistema mecánico y el subsistema eléctrico, teniendo en cuenta los valores del momento de inercia y fricción viscosa en la transmisión del engranaje reductor del motor de radio b, medido correcta - mente en forma experimental, lo que está detallado en el Anexo B, y que la varilla rota a la misma velocidad y el mismo ángulo que el engranaje re - ductor del motor. Luego, se podrá utilizar los parámetros que obtengamos utilizando el momento de inercia y fricción viscosa, medidos en el eje secundario del motor como constantes.

2.1.3 Conversión de energía eléctrica en mecánica

El amplificador del motor tiene una ganancia Kac y una tensión de entrada u que multiplicadas obtenemos la tensión de armadura como se ex - presa en la siguiente ecuación:

$$Va = Kacu \tag{2.14}$$

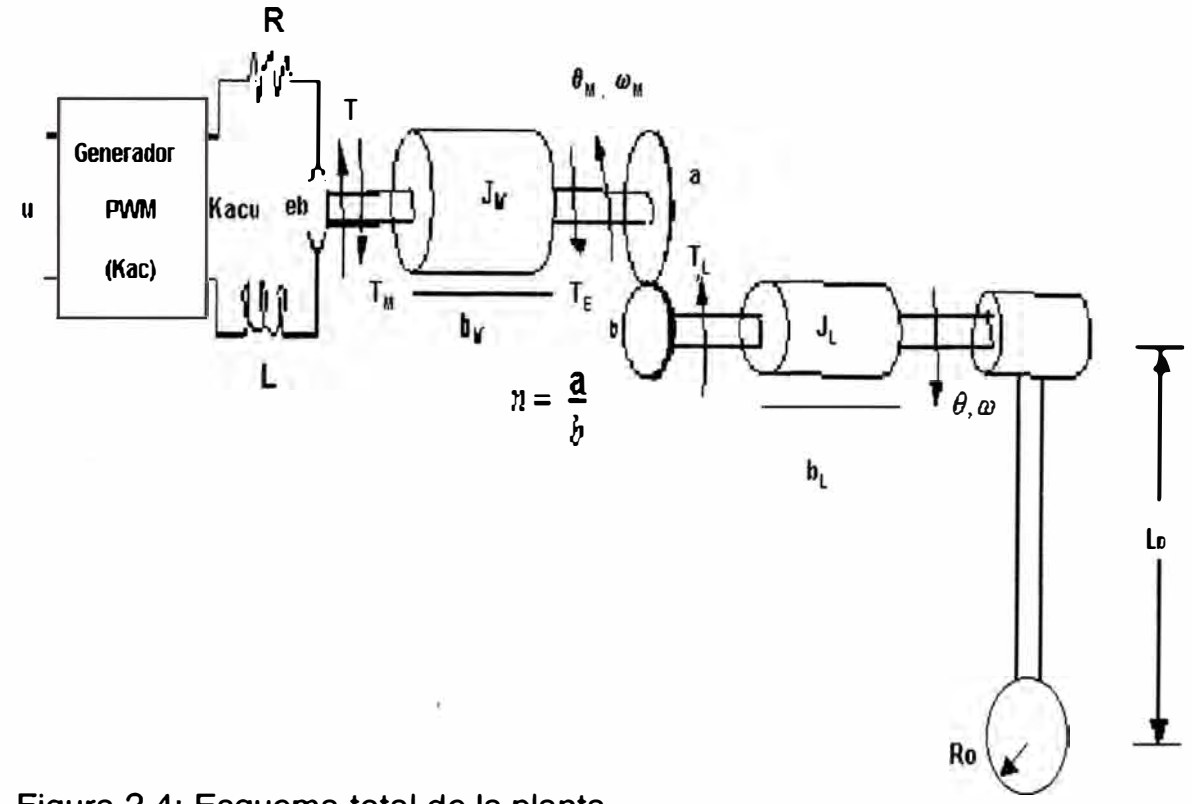

Figura 2.4: Esquema total de la planta.

Para relacionar las ecuaciones (2.8) y (2.13) tenemos utilizando el principio de la conversión de energía eléctrica en mecánica representada por la siguiente ecuación:

$$T = KtI \tag{2.15}$$

Donde Kt es la constante de par motriz. Igualando las ecuaciones (2.13) y (2.14), y como se tiene un motor de corriente continua, la corriente permanece constante, por lo que la derivada de la corriente es nula, con lo que podemos simplificar el orden de tercero a uno de segundo orden

efectuando operaciones, despejando la corriente obtenemos la ecuación (2.16):

$$I = \frac{Kacu \;-\; En}{R} \qquad (2.16)$$

Reemplazando (2.8) en (2.15), despejando la derivada de la velocidad angular y teniendo en cuenta la corriente de la ecuación (2.16) obtenemos:

$$\dot{\omega} = -\left(\frac{A}{N}\right)\operatorname{sen}\theta - \left(\frac{beff + KtEn/R}{N}\right)\omega - \frac{f(n\omega)}{N} + \left(\frac{KtKac}{NR}\right)u \qquad (2.17)$$

La ecuación (2.17) representa la ecuación de la planta no lineal, como se tiene en la referencia [16] para mayor seriedad, pudiendo formularse así:

$$\begin{aligned} \dot{x} &= F(x) + G(x)u \\ y &= H(x) \end{aligned} \qquad (2.18)$$

Eligiendo las variables de estado de la siguiente manera:

$x_1 = \theta = \ldots$ posición angular.

$x_2 = \dot{\theta} = \omega \ldots$ velocidad angular.

Obtenemos:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\left(\frac{A}{N}\right)\operatorname{sen}x_1 - \left(\frac{beff + KtEn/R}{N}\right)x_2 - \frac{f(nx_2)}{N} + \frac{KtKac}{NR}u \end{aligned} \qquad (2.19)$$

Con salida: $y=x_1$.

Usando (2.19), la expresión (2.18) puede rescribirse como:

$$F(x) = \begin{bmatrix} x_2 \\ -\dfrac{A\operatorname{sen}x_1}{N} - \dfrac{f(nx_2)}{N} - \left(\dfrac{beff + KtEn/R}{N}\right)x_2 \end{bmatrix} \qquad (2.20)$$

$$G(x) = \begin{bmatrix} 0 \\ \dfrac{KtKac}{NR} \end{bmatrix} \qquad (2.21)$$

$$H(x) = x_1 \qquad (2.22)$$

Las ecuaciones (2.20), (2.21) y (2.22) representan el modelo mate - mático de la planta, que es un sistema SISO, una entrada para una salida, considerando la carga no-lineal, se tiene un sistema de segundo orden.

2.2 Modelo matemático de la carga, sin variación de corriente

Al considerar que la corriente permanece constante por que estamos tratando con un motor de corriente continua, entonces, la derivada de la in - tensidad de corriente de armadura se hace nula, por lo que la derivada de la variable de estado x_3 desaparece, reduciéndose así el orden del sistema. Considerando la ecuación (2.16) en la que se despeja la corriente, reempla – zando la ecuación (2.8) en (2.15) obtuvimos la ecuación (2.17) y planteamos el sistema de la ecuación (2.19) de la que puede plantearse un sistema de funciones de ecuaciones de primer orden (2.23) siguiente :

$$\dot{x}_1 = x_2 = f_1(x_1, x_2, u)$$

$$\dot{x}_2 = \left(\frac{A}{N}\right)\text{sen}x_1 \left(\frac{beff + KtEn/R}{N}\right)x_2 \frac{f(nx_2)}{N} + \frac{KtKac}{NR}u = f_2(x_1, x_2, u) \quad (2.23)$$

Haciendo $\omega = \dot{\theta}$ en la ecuación (2.17) toma la forma:

$$\ddot{\theta} = -\frac{A}{N}\text{sen}x_1 - \left(\frac{beff + KtEn/R}{N}\right)x_2 - \frac{f(nx_2)}{N} + \left(\frac{KtKac}{NR}\right)u \quad (2.24)$$

Dada la fricción estática y de Coulomb por: $f(nx_2) = C1\ \text{Sgn}(n\dot{\theta})$, don - de C1 es una constante para modelar un efecto compensatorio que acepte como máximo 0.15 voltios por este efecto; tomando transformadas de La - place se tiene la siguiente ecuación (2.25):

$$s\dot{\theta} = -\frac{A}{N}\text{sen}x_1 - \left(\frac{beff + KtEn/R}{N}\right)x_2 - \frac{C1\text{Sgn}(n\dot{\theta})}{N} + \left(\frac{KtKac}{NR}\right)u \quad (2.25)$$

Ordenando la ecuación (2.25) y reemplazando $x_2 = \theta$ obtenemos la ecua - ción siguiente:

$$\dot{\theta} = \frac{\frac{1}{N}}{s + \left(\frac{beff}{N} + \frac{KtEn}{NR}\right)}\left[-Asen\theta - C1Sgn(n\dot{\theta}) + \frac{KtKac}{R}u\right] \qquad (2.26)$$

2.3 La comunicación de la planta

Básicamente el motor recibe señales y responde a esos estímulos con movimiento de su eje pero nosotros necesitamos datos que nos den infor - mación en forma de señal eléctrica del estado en cada instante de la planta esto el motor no lo hace por si solo se requiere los tres elementos que se enumeran a continuación y que se profundizarán en esta sección como son:

2.3.1 El generador por ancho de pulsos (PWM)

Este circuito en una placa impresa es el encargado de comunicar al motor de corriente continua en forma amplificada y adecuada a la posición angular que se desee la señal de control que es calculada por programa y que se obtiene de algún puerto de salida del DSP en palabras de ocho bits que son los más significativos de los veinticuatro bits de palabra con que trabaja el DSP convencionalmente. Este generador esta conformado por dos etapas que son:

i. El generador de PWM.

ii. El Amplificador PWM ó driver.

i El generador de PWM.

Se encarga de conmutar al motor mediante una serie de pulsos, logrando así, variar la velocidad de acuerdo al ancho de pulso. El LM3524 genera una modulación por ancho de pulso de decenas de Kilohertz, esta señal será amplificada por el driver antes de alimentar al motor. La figura 2.5 muestra el generador de señales PWM y PWM con zona muerta. El LM3524 es un generador de PWM, el pin 9 (comparador) recibe la señal de control de la tarjeta de adquisición de datos Lab-PC+, el rango de tensión en el pin 9 puede varia entre 0.8 a 3.7 voltios, un valor superior ó inferior satura al LM3524; la base de tiempo se controla por R_{27} y C_5 trabajándose aproxima - damente a una frecuencia de 15.4 Kilohertz; en C_A y C_B se forman dos se - ñales pulsantes desfasadas 180°, con un ciclo de trabajo limitado a variar de 0 a 50%; la suma de éstas da lugar a una señal PWM con un ciclo de trabajo variable de 0 a 100%; el efecto de R_{28} es sumar las señales de C_A y C_B; los condensadores C_3 y C_4 se utilizan para filtrar el ruido.

En la figura 2.5 se muestra el circuito generador de las señales PWM y de lógica digital que genera el circuito de disparo para la parte analógica, que son descritas a continuación por que también se muestran en la figura 2.5. En la figura 2.6 se observa el diagrama de tiempos de la generación de PWM y PWM con zona muerta, se está considerando los retardos de las compuertas lógicas utilizadas. La señal PWM se muestra en (a); PWM (b) se deriva de U6F con un retardo T_a; U5A se utiliza como seguidor, aumen - tando el FAN OUT de OSC (c), U7B y U7A son disparados con los flancos

de subida de PWM y OSC respectivamente; T_d es el retardo de U7A, el ancho del pulso de U7A-13 (d) está controlado por R30 y C7; similarmente, T_b es el retardo de U7B, el ancho del pulso de U7B-5 (g) es controlado por R29 y C6. El tiempo muerto entre el flanco de bajada de PWM y el flanco de subida de $\overline{PWM}$ está dado por U7A-13 (d) el cual es adicionado a PWM generando la nueva señal de control PWM_{OSN} = PWM+U7A-13 (f). El tiempo muerto para el siguiente flanco es dado por U7B-5 éste se adiciona a $\overline{PWM}$ generando la nueva señal de control PWM_{ON} = $\overline{PWM}$ +U7B-5 (g) en esta señal se observa un retardo T_c, esto se puede ignorar porque no será visible al conmutador el cual tiene un retardo superior a T_c que es igual al dado por una compuerta lógica.

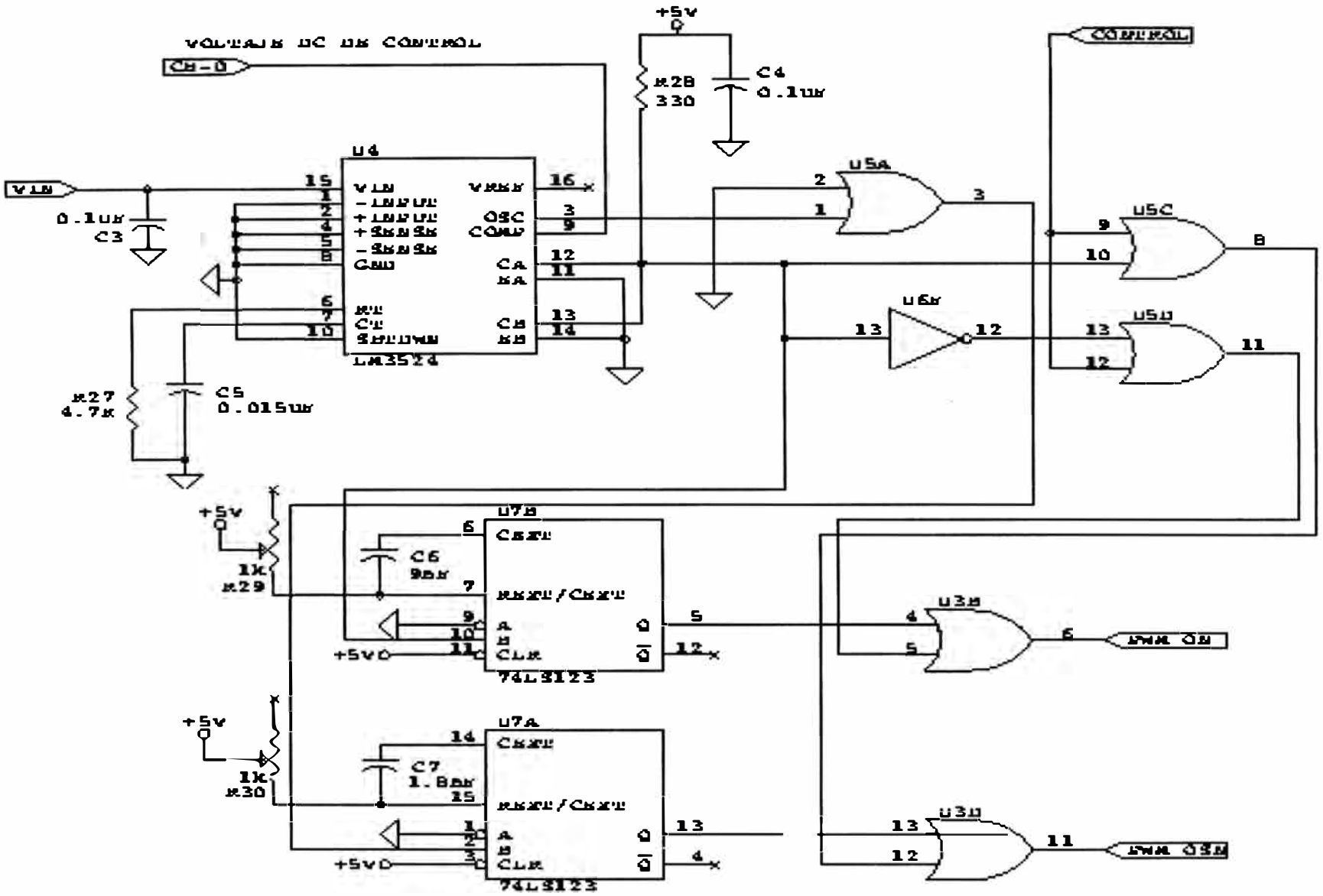

Figura 2.5: Circuito generador de PWM.

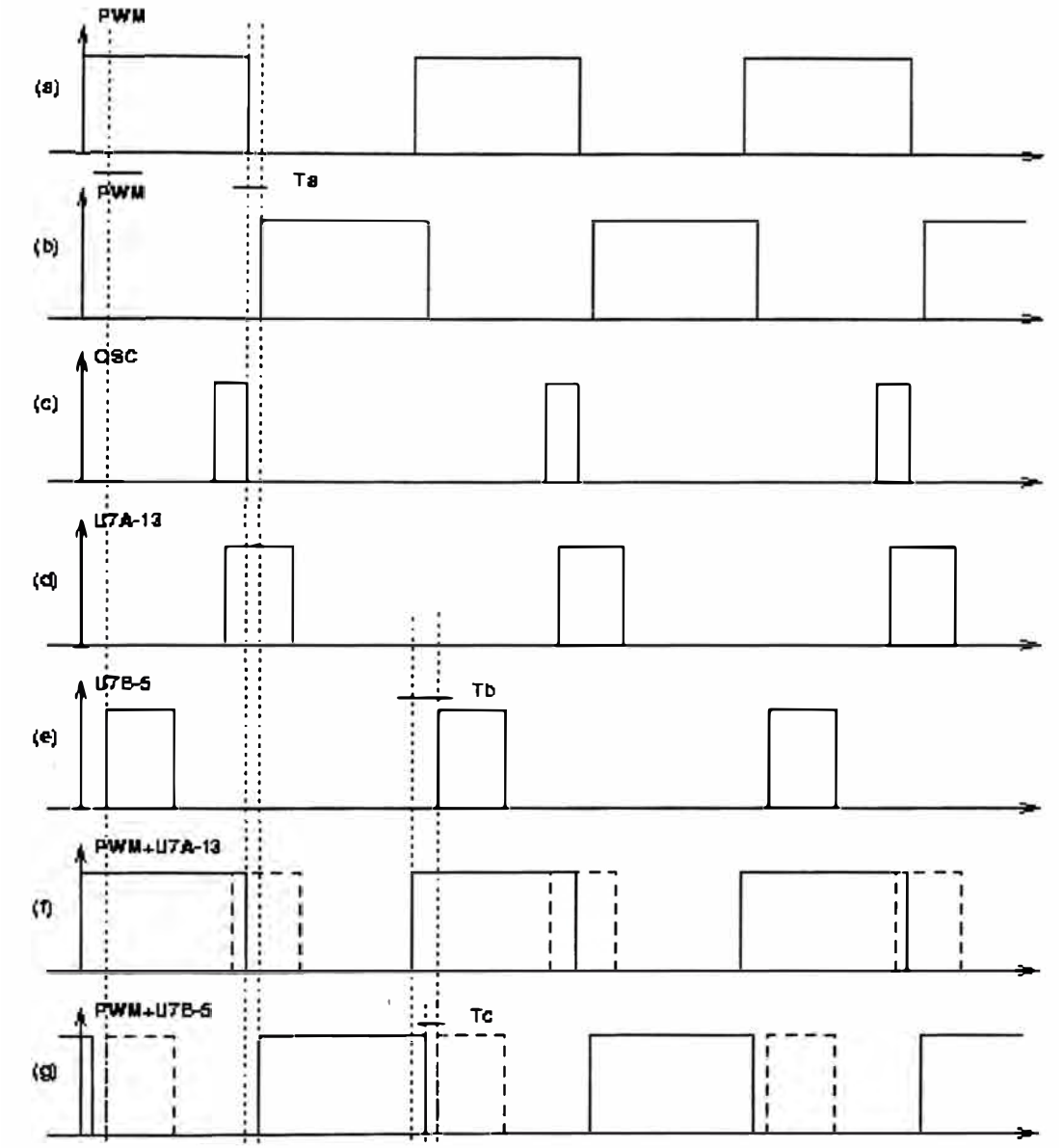

Figura 2.6:Diagrama de tiempos del circuito generador de PWM.

ii El Amplificador PWM ó driver.

En la figura 2.7 se muestra el esquema del funcionamiento del driver. Un sistema de disparo y dos pares de conmutadores A y B conforman el driver. El objetivo del sistema de disparo es realizar la conmutación de cada par A y B, de tal forma que se eviten cortocircuitos durante el tiempo de conmutación de A a B. Esto se manifiesta cuando el sistema de disparo es inadecuado, en el caso de una señal PWM y otra, las cuales controlan

compuertas de conmutadores diferentes que deberían estar complementa - dos en todo instante, el tiempo en que estos conmutadores están cerrados simultáneamente, cortocircuitan la fuente, lo que produce respuestas transi - torias subamortiguadas durante el tiempo de conmutación, lo que se atenúa con un condensador adecuado, el efecto perjudicial se produce cuando en el tiempo en que los dos conmutadores se encuentran cerrados circula una corriente significativa que va deteriorando al dispositivo conmutador.

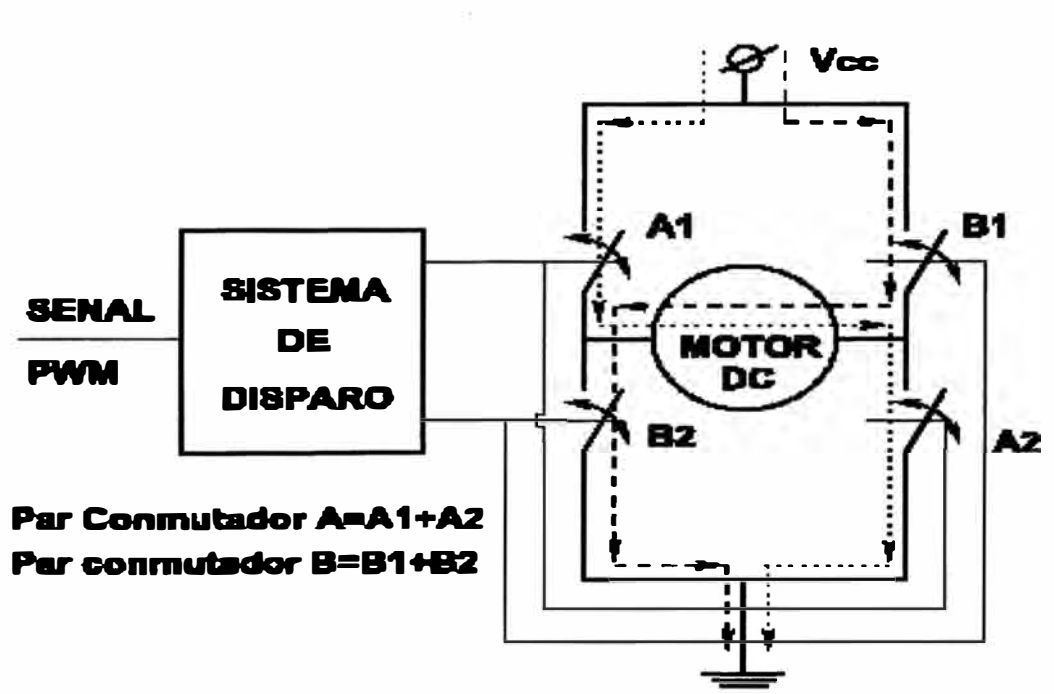

Figura 2.7: Esquema general del sistema de disparo y los conmutadores. Cuando el sistema de disparo cierra el conmutador A y abre el B, el sentido de la corriente es la línea con puntos, induciendo de esta forma una tensión +Vcc en el motor. Luego, abre el conmutador A y cierra el B. El sentido de la corriente es la línea segmentada, induciendo así una tensión –Vcc en el mo - tor. Por consiguiente el motor ve en sus bornes una onda de voltaje cuadra - da, variando entre ±Vcc y la corriente que puede absorber dependerá de los conmutadores.

Circuito amplificador PWM

En la figura 2.8 se muestra la etapa amplificadora de PWM_{ON} y de PWM_{OSN}, su funcionamiento y un esquema general es el mostrado en la fi - gura 2.7. Un conmutador puede ser hecho de varias formas, una cambiando la frecuencia de conmutación ó el ciclo de trabajo y otra cambiando ambas. El efecto buscado en un método u otro es que el promedio de voltaje de salida sea proporcional a la tensión de comando. El método más común de un conmutador amplificador es la modulación por ancho de pulso (PWM). Aquí los dispositivos conmutadores mosfet's, conmutan a una frecuencia constante, dando como resultado la variación de la tensión de salida entre dos valores extremos. Por variación del ancho de pulso ó el ciclo de trabajo, el valor promedio de voltaje de salida puede ser cambiado en forma proporcional a la tensión de comando.

El conmutador amplificador implementado es de tipo "H" su diagrama es visto en la figura 2.7. La principal ventaja del tipo "H" es que sólo es ne - cesario una fuente y son requeridos cuando se necesita una alta tensión; los dispositivos conmutadores son conmutados en pares generando una tensión bipolar a la salida. El efecto de tiempo muerto, el cual es explicado en la etapa de disparo consiste en dejar al motor en el “aire”, como el motor tiene una inductancia trata de mantener la corriente aumentando la tensión. Por consiguiente, hay que tener cuidado en la elección de la duración del tiempo muerto, minimizando su efecto, siendo el típico entre 1 y 3 microsegundos. El amplificador tipo “H” es implementado con dispositivos conmutadores mosfet's Buz 80 y una fuente de 24 voltios; para disparar los mosfet's es

necesaria una circuitería adicional para que genere la tensión de disparo en cada conmutador. En la figura 2.8, el circuito de disparo de Q14 consiste de dos transistores Q1 y Q3 forman un amplificador que trabaja en clase B, este amplificador es manejado por un una compuerta NOT de colector abierto que dependiendo de la señal PWM_{OSN} se llega a saturar ó cortar a Q1, el fin de Q3 es absorber las corrientes de fuga de Q14. El circuito de disparo Q5 consiste también de un amplificador que trabaja en clase B el cual está formado por Q2 y Q4; como la tensión de disparo está dada por V_{GS} (diferencia de tensión entre los pines gate y source del mosfet) y donde V_S no es tierra, entonces, Q2 es alimentado por el condensador C1 el cual tiene como referencia a V_S, es decir, C1 juega el papel de fuente; el diodo D1 con la fuente de 12Voltios dan la tensión y corriente a C1 y al circuito amplificador. Q12 sirve para llevar a corte ó saturación a Q2, Q12 es a su vez llevado a corte ó saturación por la señal PWM_{ON} que pasa por dos compuertas NOT de colector abierto, Q4 sirve para absorber las corrientes de fuga de Q5. El circuito de disparo de Q6 y Q7 son idénticos al de Q5 y Q14, respectivamente.

Amplificador de Potencia del PWM

Para analizar el comportamiento del amplificador se asume que el dri - ver actúa como un amplificador lineal de ganancia Kac; se puede verificar lo anterior construyendo una función de transferencia de voltaje de la tensión de salida versus la tensión de entrada en el driver, tomando un número sufi - ciente de puntos para realizar una gráfica de la cual se deduce que el Kac=14.9.

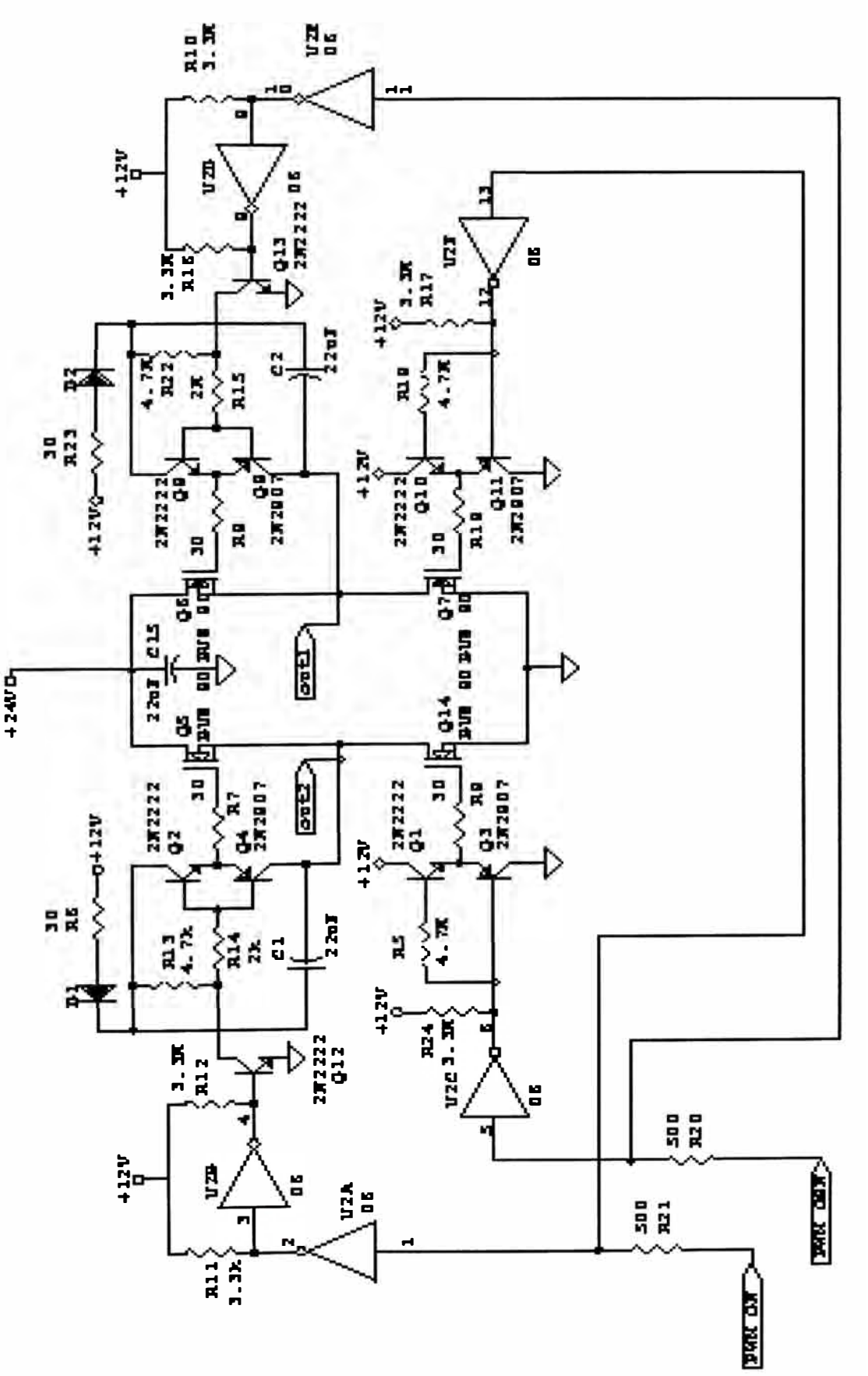

Figura 2.8: Circuito amplificador PWM.

Compensación de la fricción estática y de Coulomb

Asumiendo un modelo simplificado del sistema driver-motor, para el cual se considera la corriente como constante, esto se justifica desde la sec - ción 2.1. La ecuación (2.18) puede rescribirse no considerando la masa de la esfera y despreciando el torque T_E como:

$$Jeff\,\dot{\omega} = (\frac{KtKac}{R})u - (beff + nEKt/R)\omega - f(n\omega) \qquad (2.27)$$

Donde f(nω) modela el torque de fricción estática y de Coulomb; se puede deducir una primera aproximación de f(nω) de acuerdo a como se manifiesta la velocidad a la entrada de tensión del driver, la cual esta dado por: $f(nx_2) = C1Sgn(n\omega)$.

Cuando el motor está en eminente movimiento el torque de fricción toma distintos valores (fricción estática); si la velocidad es distinta de cero el torque toma dos valores constantes dependiendo del sentido de giro del mo - tor (fricción de Coulomb).

De la respuesta velocidad angular del motor versus la tensión de en - trada en el driver; se tiene que la zona muerta es observada cuando la velo - cidad es cero, esto es debido a la fricción estática en el motor. Una pequeña tensión de realimentación es adicionada a la entrada para compensar los distintos valores que toma el torque de fricción cuando la velocidad es cero. Como la fricción estática ocurre a bajas velocidades cuando el motor está a punto de detenerse; la realimentación también tiene que ser aplicada cuan - do la velocidad del motor se encuentre dentro de un umbral alrededor de cero.

Para linealizar añadir la tensión de compensación a la entrada del driver en la dirección del movimiento para reducir la fricción de Coulomb y el efecto de zona muerta, se debe de identificar el valor de C1 y Kac para de - terminar la tensión que se tiene que adicionar. En el caso de control de posi - ción cuando hay movimiento y/o velocidad. Sin embargo, C1 es pequeña en motores de corriente continua, siendo difícil su identificación en presencia de ruido. Por lo tanto se ajustó al valor máximo (peor caso).

La justificación de una tensión de compensación realimentada está basada en la dependencia del modelo de fricción con la velocidad. La fric - ción estática es la responsable de mantener en reposo al motor, hasta que se excede un umbral de movimiento eminente; esta tensión de compensa - ción equilibra el torque resultante de una corriente aplicada. El propósito de la tensión de compensación es mantener el punto de operación del motor en la zona de eminente movimiento, dependiendo de la dirección de la corriente aplicada. Cuando se inicia el movimiento, la fricción de Coulomb y la fricción de viscosidad actúan juntas oponiéndose al movimiento del motor. El efecto no lineal de la fricción de Coulomb es reducido al aplicar una tensión de compensación adecuada dependiendo de la dirección del movimiento. El resultado obtenido es una aproximada dependencia lineal de la velocidad con la fricción.

2.3.2 El codificador óptico y PLD(sensores)

Este circuito en una placa impresa que cuenta los pulsos que le envía el sensor del codificador óptico y que indican la posición angular del eje pri - mario del motor en cada instante como respuesta al estimulo producido por la señal recibida por el motor del generador por ancho de pulsos, asociando 512 ranuras en el disco del sensor con un ángulo de 2π radianes, para obte - ner el ángulo girado, se divide el ángulo girado por el motor entre la reduc - ción n de 19.741, el contador conformado por circuitos TTL de circuito inte - grado constituyen ocho bits de palabra y que su valor depende de la ponde – ración en voltaje del bit menos significativo del contador, recordando que la tarjeta Lab-PC+ comunica estos ocho bits del contador de pulsos del codifi - dor óptico a la microcomputadora, dicha ponderación debe ser entre 0 y 1 para un único sentido de giro de la posición angular del eje del motor y que debe coincidir con el valor en radianes de la posición angular medida, cuan - do se convierte dicha señal digital de ocho bits del contador de pulsos al va - lor análogo de continua en el rango de 0 a 1 voltios coincidentes con el valor en radianes de la posición angular medida, esto por que el DSP trabaja usualmente por precisión con números fraccionarios. El sensor de posición y sentido de giro del motor cuenta con dos entradas de alimentación y dos salidas seriales de tren de pulsos, desfasados 90º. Básicamente el codifica - dor óptico rotatorio (sensor) consiste de 4 componentes los cuales son:

1. Una fuente de luz

- El cual puede ser un diodo emisor de luz (LED) ó una lámpara incandescente. Esto depende de las consideraciones de diseño.

2. Un disco opaco con segmentos transparentes

• Localizado entre la fuente de luz y su asociado sensor de luz, este disco se coloca dé tal manera que siga el movimiento del dispositivo a ser sensado; la resolución del disco utilizado es de 512 pulsos por revolución para indicar el ángulo variado en el último periodo de discretización.

3. Un sensor de luz, frecuentemente se utiliza un fototransistor.

4. Un circuito que convierta la salida del sensor a un conveniente formato de información necesaria para la interface (PLD) usada. En la figura 2.9, se muestra un esquema del funcionamiento de un codificador óptico rotatorio.

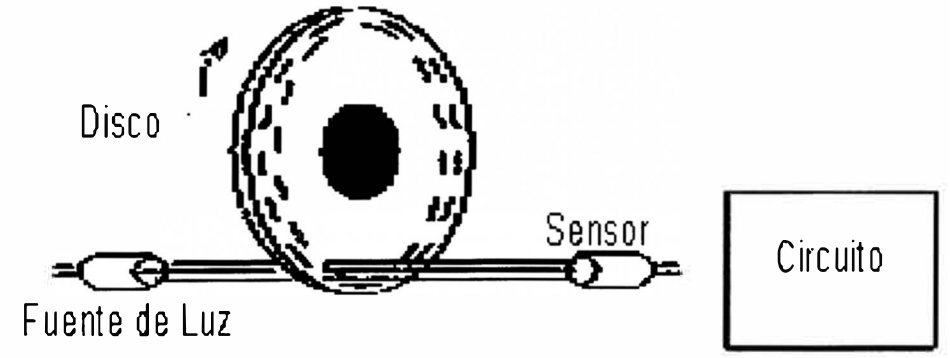

Figura 2.9: Esquema general del codificador óptico rotatorio.

Un PLD (Dispositivo lógico programable) es un circuito integrado digi - tal, configurable por el usuario; es utilizado para implementar expresiones Booleanas ó funciones que se han construido con estructuras lógicas. Esta es la principal diferencia con respecto a los dispositivos lógicos tales como los TTL, que proveen una especifica función lógica y no pueden ser modifi - cados. En un EPLD (PLD que utiliza un EEPROM como elemento reconfigu - rable) EPM7128E de la familia MAX 7000 se ha integrado un circuito digital sensor de posición que consiste de un LS7083 y 4 contadores TTL LS74193.

2.3.3 La tarjeta de adquisición de datos

Para poder revisar el trabajo de comunicación de los elementos del sistema debe tenerse en cuenta el trabajo de la tarjeta Lab-PC+ utilizada en el sistema que se tiene en el laboratorio en que dicha tarjeta está instalada en una microcomputadora Pentium, de la referencia [1] se tiene la descrip - ción de los pines por ejemplo para la salida análoga DAC0 OUT (pin No 10) que es donde se obtiene la señal de control u pues la tarjeta está comunica - da con la PC que es la que calcula dicha señal con un programa normal - mente en lenguaje C++, dicha señal puede variar de cero a diez voltios con ajuste de un potenciómetro de los datos digitales binarios del resultado dado por la computadora, por ejemplo representar diez voltios como la escritura en el convertidor digital para analógico del número 4095. Asimismo el uso del dispositivo 8255A que representan para la tarjeta veinticuatro pines pro - gramables de entrada y salida. Estos pines representan los puertos de entrada y salida de ocho bits (A, B, y C) respectivamente. Estos puertos pueden ser programados como dos grupos de señales de doce bits ó como tres puertos individuales de ocho bits cada uno. Respetando la filosofía de la tarjeta Lab-PC+, pues los puertos paralelos del DSP tienen la señal digital de ocho bits que son los más significativos de la palabra de veinticuatro bits, se puede utilizar un dispositivo convertidor para analógico (DAC) externo también de ocho bits, obteniendo una señal analógica a la cual hay que agregarle la señal continua de offset de 2.3 voltios para enviar la señal resultante a la entrada del generador PWM.

2.4 El procesador digital de señales DSP56002

El procesamiento digital de señales se puede definir como el proce - samiento aritmético de señales en tiempo real, digitalizadas y muestreadas a un intervalo regular de tiempo. Tradicionalmente las funciones efectuadas mediante el DSP (anacrónico de procesamiento digital de señales ó proce - sador de señales, dependiendo del contexto) eran realizadas mediante el uso de circuitos analógicos, trayendo consigo ventajas y desventajas ya co - nocidas. En la actualidad, gracias al avance de la tecnología de los semi - conductores ha sido posible construir procesadores digitales de señales que efectúen las mismas funciones de manera más eficiente, transformando el problema de hardware a uno de software. Hay que resaltar que en esta tesis se realizan experiencias usando DSP de 24 bits, utilizando su propio len - guaje ensamblador, en esta versión de DSP se cuenta con un registro de PLL, que permite aumentar ó disminuir la frecuencia de operación de 4 Me – gahertz. La tarjeta DSP56002EVM es una plataforma, diseñada para fami - liarizar al usuario con el procesador digital de señales DSP56002 como se tiene en la figura 2.10.

Los 24 bits de precisión del DSP, los 32K Word de sram externa hacen la tarjeta ideal para implementar y demostrar algoritmos, así como para el aprendizaje de la arquitectura y el conjunto de instrucciones del procesador DSP56002. Para utilizar la tarjeta, el usuario necesita como mínimo una fuente de alimentación de 7-9 Voltios, 700mA, un cable serial RS-232, y una IBM PC compatible 386 ó superior, utilizamos Pentium.

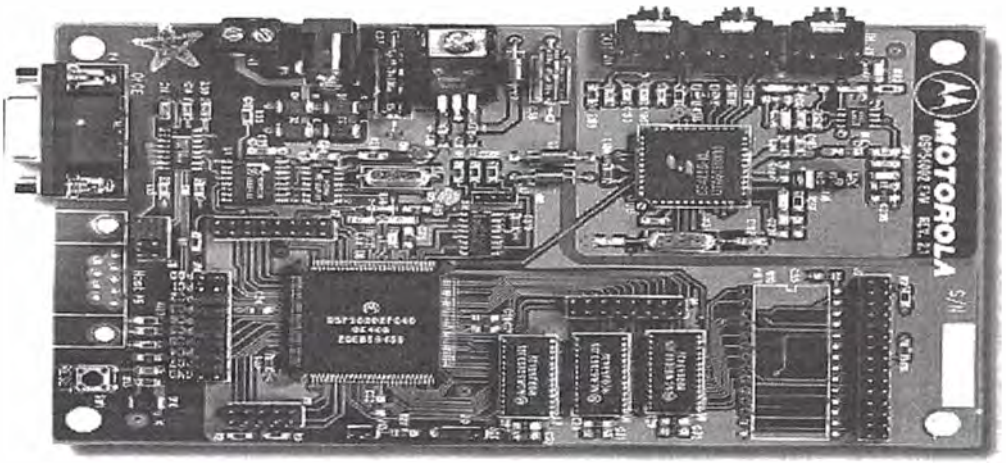

Figura 2.10: La tarjeta de desarrollo del DSP56002EVM.

Características de hardware utilizadas del DSP56002EVM son:

1. Opera a 40 MHZ, más de 20 millones de instrucciones por segundo, y más de 120 millones de operaciones por segundo.

2. Cuatro buses de 24 bits de data y tres buses de 16 bits de direcciones para acceso a una memoria de programa (P), y dos memorias de data (X, Y).

3. Memoria de programa de 512x24 bits (memoria P).

4. Dos memorias ram de data 256x24 bits (memoria X, Y).

5. Temporizador y contador de eventos de 24 bits, para generar y medir ondas digitales.

6. PLL programable por software.

7. Un puerto (once) para la depuración y ejecución de programas mediante PC.

8. Un regulador de voltaje (MC7805).

9. Interface serial RSS232-once.

10. Microcontrolador MC68HC705K1 para conversión de comandos del RS232 a once.

2.4.1 Registros del DSP56002

El módulo de procesamiento central del DSP56002 está organizado alrededor de los registros de las 3 unidades existentes que son:

1. Registros de la Unidad Aritmética de Datos.

La unidad aritmética de datos está compuesta por 4 registros de pro - pósito general de 24 bits: X0, X1, Y0, Y1; los cuales pueden ser tratados en forma independiente como registros de 24 bits ó como registros de 48 bits, en cuyo caso serán llamados simplemente X, Y. Estos registros de propósito general nos sirven como buffers de entrada entre el bus de datos X ó Y con la unidad multiplicadora y acumuladora (MAC).

También existen dos acumuladores de propósito general de 56 bits: A, B. Cada uno esta formado por 3 registros concatenados: A2, A1, A0 ó B2, B1, B0. Los registros A2 y B2 son de 8 bits y son usados cuando se requiere más de 48 bits de precisión, A1 y B1 son de 24 bits y almacenan los 24 bits más significativos del producto (MSP), A0 y B0 también son de 24 bits y almacenan los 24 bits menos significativos de producto(LSP).

2. Registros de la Unidad Generadora de Direcciones

La unidad Generadora de Direcciones posee dos unidades aritméti - cas que pueden generar simultáneamente 2 direcciones de 16 bits. Posee tres tipos de registros:

(a) Address Register Files(Rn):

Cada uno de los cuales está formado por 4 registros de 16 bits: R0, R1, R2, R3 y R4, R5, R6 y R7, que almacenan direcciones ó data de propó - sito general.

(b) Offset Register Files(Nn):

Cada uno de los cuales está formado por 4 registros de 16 bits: N0, N1, N2, N3 y N4, N5, N6 y N7, que contiene los valores de offset usados para incrementar ó decrementar las direcciones y pueden ser usados como registros de 16 bits de propósito general.

(c) Modifier Register Files(Mn):

Son 8 registros de 16 bits: M0, M1, M2, M3 y M4, M5, M6 y M7, usados para definir el tipo de dirección aritmética a realizar en los cálculos de modos de dirección, ó como registros de 16 bits de propósito general.

3. Registros de la Unidad de Control de Programa

Son los registros que nos indican el estado actual del DSP. Cada uno de los cuales tiene una función particular.

•**Contador de Programa:** Registro de 16 bits que contiene la direc – ción de la próxima locación a ser alcanzada en la memoria de programa.

•**Status Register:** Registro de 16 bits que contiene dos registros internos: el mode register (MR) y el condition code register (CCR). Los cuales definen el estado actual del sistema y del usuario del procesador.

•**Operating Mode Register:** Registro de 24 bits que configura el mo - do de operación actual del procesador.

•**Loop Counter Register:** Registro de 16 bits el cual especifica el número de veces que debe ser repetido un lazo de programa.

•**Stack Pointer Register:** Indica la locación del system stack, es referenciado implícitamente por algunas instrucciones ó directamente por la instrucción MOVEC.

2.4.2 Puertos del DSP

1. Puerto A

El puerto A del DSP56002EVM nos ofrece una forma versátil de poder acceder a una ó más de sus memorias principales (X, Y y P) mientras ejecu - ta una instrucción pues su uso es fundamentalmente para correr el progra - ma, sin embargo cuando una ó más de las memorias es externa, el acceso puede requerir más de un ciclo de instrucción, pues sólo es posible un acce - so a una memoria externa a la vez.

La información de este puerto del DSP, se puede hallar en el Capítulo 4 de la referencia [6], y el capítulo 8 de la referencia [5].

2. Puerto B

El puerto B del DSP56002 puede ser configurado para funcionar como puerto de propósito general de entrada / salida por medio de 15 pines, ó como host Interface bidireccional de 8 bits.

Cuando es configurado como puerto de entrada / salida de propósito general, el puerto B es gobernado por tres registros: de control, data y de di - rección, que se tienen a continuación,

(a) El registro de control del puerto B (PBC).

- Se encuentra ubicado en la locación $FFE0 de la memoria de data X (X: $FFE0), En este registro se especifica si es que el puerto B fun - cionará como host Interface ó puerto de entrada / salida de propósito general. Sólo 2 bits de los 24 son configurables.

(b) El registro de dirección de data del puerto B (PBDDR).

- Se encuentra ubicado en la locación $FFE2 de la memoria de data X (X: $FFE2), En este registro se determina la condición de entrada ó salida de cada uno de los pines del puerto B; puesto que se disponen de 15 bits, entonces se configurarán sólo 15 bits del registro de 24.

(c) El registro de data del puerto B (PBD).

- Se encuentra ubicado en la locación $FFE4 de memoria de data X (X: $FFF4). Este registro almacena los valores lógicos que corres - ponden a cada uno de los 15 bits existentes.

Físicamente el Puerto B se encuentra en el módulo J7 de la tarjeta. La información de este puerto se tiene en el Capítulo 5 del manual del usurario en la referencia [6].

3. Puerto C

El puerto C del DSPP56002 es un puerto de nueve pines que puede ser usado de dos formas:

(a) Tres de los nueve pines pueden ser configurados como entrada / salida de propósito general, ó como pines de interface de comunicación serial (SCI) como otra opción.

(b) Los otros 6 pines pueden ser configurados como entrada / salida de propósito general, ó como pines de interface serial síncrona (SSI) como la otra opción.

Cuando el puerto C es configurado como puerto de entrada / salida de propósito general, puede ser usado como dispositivo de control. Cuando es usado como interface serial, el puerto C nos brinda una forma conveniente

de conexión con otros procesadores, convertidores A/D y D/A y muchos tipos de transductores.

Cuando es configurado como puerto de entrada / salida de propósito general, el puerto C es gobernado por tres registros:

(a) El registro de control del puerto C (PCC).

- Se encuentra ubicado en la locación $FFE1 de la memoria de datos X (X: $FFE1), en este registro se especifica si es que el puerto C funcionará como interface serial síncrona (bits del 3-8), interface de comunicación serial (bits del 0-2), ó como entrada / salida de propósito general (GPIO) (bits del 0-8) como otra opción. Si el valor de un bit es cero, entonces hablaremos del puerto C como GPIO; si es uno estaremos hablando del puerto C como interface serial.

(b) El registro de dirección de data del puerto C (PCDDR).

- Se encuentra ubicado en la locación $FFE3 de la memoria de datos X (X: $FFE3), en este registro se determina la condición de entrada ó salida de cada uno de los 9 pines del puerto.

(c) El registro de data del puerto C (PCD).

- Se encuentra ubicado en la locación $FFE5 de la memoria de datos X (X: $FFE5), este registro almacena los valores lógicos que corresponde a cada uno de los 9 bits existentes.

El puerto C se encuentra físicamente en el módulo J10 de la tarjeta.

2.4.3 Temporizador y contador de eventos

El Temporizador y contador de eventos físicamente está representado por medio de un pin (TIO) ubicado en el módulo J11 de la tarjeta. Este puede ser usado como entrada ó salida; en el primer caso el TIO funciona como contador de eventos externos ó midiendo el ancho de pulso de señales periódicas también externas. En el segundo caso el módulo funciona como temporizador. Al igual que el puerto A, B y C, el temporizador y contador de eventos es gobernado por registros: el registro de estado/control del temporizador y el registro de cuenta los cuales se describen a continuación.

- El registro de estado/control de temporizador (TCSR) se encuentra ubicado en la locación $FFDE de la memoria de datos X (X: $FFDE), en este registro de 24 bits se controla y verifica el estado del temporizador. 11 de los 24 son configurables.

- El registro de cuenta del temporizador (TCR) se encuentra en la locación $FFDF de la memoria de datos X (X: $FFDF), en este registro se almacena el valor a ser cargado en el contador.

Existen 7 modos de operación del temporizador y contador de even tos, los cuales son configurados por medio del registro de estado/control.

2.4.4 El oscilador por enganche de fase (PLL)

El PLL Clock Oscillator es parte del módulo de procesamiento central del DSP56002, permite al procesador operar a una alta frecuencia de reloj interno usando un reloj de entrada de baja frecuencia. Esta característica ofrece dos beneficios inmediatos:

- La baja frecuencia del reloj de entrada reduce el efecto de interfe - rencia electromagnética generada por el sistema.

- La capacidad de oscilar a diferentes frecuencias reduce los costos, al no tenerse la necesidad de adicionar osciladores al sistema.

El PLL realiza multiplicación de frecuencias para permitir al procesa - dor usar los relojes del sistema externos disponible a máxima velocidad de operación. Está gobernado por un registro de control PCTL de 24 bits ubica - do en la locación $FFFD de la memoria de datos X (X: $FFFD), este registro está dividido en 8 grupos de bits con funciones diferentes, los principales son los siguientes:

- Bit de Factor de Multiplicación (MF): Bits 0-11. Definen el factor a multiplicar a la frecuencia de entrada del PLL. Puede ser un entero contenido entre 1 y 4096.

- Bit de Factor de División (DF): Bits 12-15. Definen el exponente de la potencia del divisor (2), su valor varía entre 0 y 15.

- Bit de deshabilitación de cristal (XTLD): Bit 16. Controla la salida del cristal oscilador. Se recomienda que este bit este configurado (Cristal deshabilitado) para reducir el ruido y la potencia de disipación.

• Bit de habilitación del PLL (PEN): Bit 18. Cuando este bit está configurado, el está habilitado y los relojes internos serán derivados de la salida del PLL VCO. En caso contrario el PLL está deshabilitado y los relojes internos son derivados directamente del reloj conectado al pin XTAL.

2.5 Implementación de la interface para comunicación con el gene - rador PWM

El circuito debe agregar la señal de offset de 2.3v a la señal de control en un puerto paralelo del DSP (B ó C) con por lo menos 8 bits que utiliza un convertidor digital para analógico, la figura 2.11, muestra el circuito que lo hace.

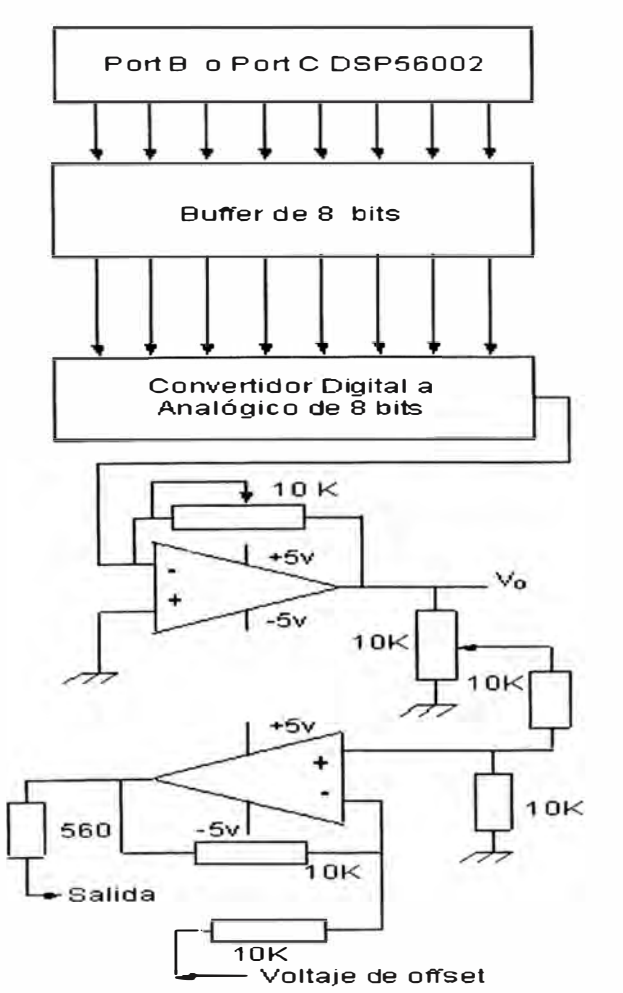

Figura 2.11: Implementación para comunicación con el generador PWM.

CAPÍTULO III
LINEALIZACIÓN Y DISCRETIZACIÓN

Obtenidas las ecuaciones del modelo que representa la planta en el capítulo previó procedemos a obtener los parámetros que identifican a nuestra planta asumiendo un sistema de segundo orden, primero se hacen las consideraciones de linealización para lo cual eliminamos expresiones de segundo grado basando nuestro estudio en la línea recta dentro de peque - ños rangos en donde podemos suponer que nuestros parámetros permane - cen constantes.

3.1 Linealización del modelo no-lineal

De las ecuaciones matriciales (2.20), (2.21) y)2.22), para obtener un modelo matemático lineal de un sistema no lineal, se supone que las varia - bles se desvían levemente de alguna condición de operación. Esta aproxi - mación lineal se puede obtener mediante la expansión en series de Taylor despreciando los términos de orden superior.

Considerando que nuestro sistema está representado por la siguiente ecuación:

$$\dot{\vec{X}} = \vec{f}\left(\vec{X}, U(t), \vec{u}(\vec{X}, t), t \right) \qquad (3.1)$$

Donde: X (de orden 2x1) y U(t) (de orden 1) son el vector de estados y la ley de control respectivamente. La función $\vec{f}(.)$ puede también tener pertur - baciones $\vec{u}(.)$ en los estados. De la ecuación (3.1), es posible obtener la siguiente representación de modelo lineal:

$$\dot{\vec{X}}(t) = [A_C + \Delta A_C]\vec{X}(t) + [B_C + \Delta B_C]U(t) + \vec{u}(\vec{X},t)$$
$$Y(t) = [C_C + \Delta C_C]\vec{X} + \omega(\vec{X},t) \qquad (3.2)$$

Donde el subíndice c es para tiempo continuo, A_c es la matriz de estado de orden 2, B_c es la matriz de control de orden 2x1, C_c es la matriz de salida, ΔA_c, ΔB_c y ΔC_c son las correspondientes tolerancias con adecuadas dimensiones, $\vec{u}(\vec{X},t)$ es el vector no lineal de perturbación de estados de orden 2x1, $\omega(\vec{X},t)$ es la perturbación no lineal de salida de orden 1, e Y(t) es la salida del proceso de orden 1. El proceso nominal puede ser obtenido haciendo todas las tolerancias y las perturbaciones en la ecuación (3.1) iguales a cero. Para cada instante de muestreo un modelo nominal, puede ser obtenido con la siguiente representación:

$$\vec{X}(k+1) = A\vec{X}(k) + BU(k)$$
$$Y(k) = C\vec{X}(K) \qquad (3.3)$$

Donde k es el tiempo discreto. Empleando las relaciones siguientes: $x = X - X^0$, $y = Y - Y^0$ y $u = U - U^0$, donde el superíndice o denota un valor promedio, podemos obtener las ecuaciones (3.4) en la página siguiente,

$$\vec{x}(k+1) = A\vec{x}(k) + Bu(k)$$
$$y(k) = C\vec{x}(k) \qquad (3.4)$$

De esta ecuación (3.4) se obtiene el polinomio de la forma siguiente:

$$A(z^{-1})y(z) = B(z^{-1})u(z) \qquad (3.5)$$

Donde z es el operador de la transformada z para el caso discreto; los polinomios en la ecuación (3.5) tienen la siguiente forma:

$$A(z^{-1}) = 1 + a_1 z^{-1} + \ldots + a_n z^{-n}$$
$$B(z^{-1}) = b_1 z^{-1} + \ldots + b_n z^{-n}$$

3.2 Discretización del modelo linealizado

Obtenido el modelo lineal del sistema será necesario su discretización para poder aplicarle las herramientas de análisis y control en tiempo discre - to. Si las ecuaciones de estado en tiempo continuo están dadas por:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = x(t) + Du(t) \qquad (3.6)$$

y considerando que la solución de estas últimas ecuaciones está dada por:

$$x(t) = \phi(t - t_0)x(t_0) + \int_{t_0}^{t} \phi(t - \tau)Bu(\tau)\partial\tau \qquad (3.7)$$

Donde t_0 es el tiempo inicial, y $\phi(t\text{-}t_0)$ viene dado por la ecuación (3.8):

$$\phi(t - t_0) = e^{A(t-t_0)} = \sum_{k=0}^{\infty} \frac{A^k (t - t_0)^k}{k!} \qquad (3.8)$$

Asumiremos que la entrada u alimenta a un retenedor de orden cero, de tal forma que a la salida se obtendrá una señal constante en el intervalo de dos instantes de muestreo consecutivos, es decir:

$$u(t) = u(kT), \text{ para } \quad kT \le t \le k(T+1) \qquad (3.9)$$

Para obtener el modelo en tiempo discreto evaluamos la ecuación (3.6) en el instante t = k(T+1) con $t_0 = kT$, lo que conduce a:

$$x[k(T+1)] = \phi(T)x(kT) + u(kT) \int_{kT}^{k(T+1)} \phi[T(k+1)-\tau]B\,\partial\tau \qquad (3.10)$$

Observar que hemos reemplazado u(τ) por u(kT) dado que según la ecuación (3.9) tenemos: u(t)=u(kT) para kT≤ t < k(T+1) y por lo tanto podemos hacer: u(τ)=u(kT).

Como el modelo en tiempo discreto está dado por:

$$x[k(T+1)] = Gx(kT) + Hu(kT) \qquad (3.11)$$

Comparando la ecuación (3.10) con (3.11):

$$G = \phi(T)$$

$$H = \left[\int_{kT}^{k(T+1)} \phi(kT+T-\tau)\partial\tau\right]B \qquad (3.12)$$

La integral para B de la ecuación (3.12) puede simplificarse haciendo λ=k(T+1)-τ:

$$\int_{kT}^{k(T+1)} \phi(kT+T)\partial\tau = -\int_{T}^{0} \phi(\lambda)\partial\lambda = \int_{0}^{T} \phi(\lambda)\partial\lambda \qquad (3.13)$$

Así obtenemos que:

$$H = \left[\int_{0}^{T} \phi(\lambda)\partial\lambda\right]B \qquad (3.14)$$

La ecuación de salida en tiempo discreto estará dada por:

$$y(kT) = Cx(kT) + Du(kT) \qquad (3.15)$$

En la ecuación (3.15) C y D son constantes y no dependen del perio - do de muestreo Ts, por lo tanto son las mismas que para el caso continuo. Discretizando el modelo matemático de las ecuaciones (2.20), (2.21) y (2.22), considerando que la corriente de armadura permanece constante y la masa de la esfera es igual a cero para tener el modelo del sistema en el eje secundario del motor en que Jeff y beff son tales como se detalla en el Anexo B. A partir de (2.20), (2.21) y (2.22) se obtiene la siguiente ecuación:

$$Jeff\,\dot{\omega} = \left(\frac{KtKac}{R}\right)u - \left(beff + \frac{nEKt}{R}\right)\omega - \frac{gLom}{2n}\,sen\theta - f(n\omega) \qquad (3.16)$$

En la ecuación (3.16) se tiene dos no linealidades: senθ y f(nω); rescribiendo la ecuación (3.16) en una forma más conveniente y despejando la velocidad angular del eje secundario del motor se obtiene:

$$\dot{\omega} = -\left(\frac{beff + \frac{nEKt}{R}}{Jeff}\right)\omega + \frac{KtKac}{RJeff}\left[u - \frac{RgLom}{2nKTKac}\,sen\theta - \frac{R}{KtKac}\,f(n\omega)\right] \qquad (3.17)$$

En la ecuación (3.17) se muestra que las no linealidades han sido trasladas hacia la tensión de entrada; esto es útil porque se tiene acceso directamente a u. Por lo tanto, se puede operar sobre esta variable para eliminar mediante una realimentación de compensación las no linealidades del motor.

Discretizando la ecuación (3.17) se obtiene:

$$\omega_{k+1} = \left[1 - \frac{Ts\left(beff + \frac{nEKt}{R}\right)}{Jeff}\right]\omega_k + \frac{TsKtKac}{RJeff}\left[u - \frac{RgmLo}{2nKtKac}\,sen\theta - \frac{RCl}{KtKac}\,Sgn(n\omega_k)\right] \qquad (3.18)$$

La ecuación (3.18) presenta el modelo matemático discreto del siste - ma que considera el eje secundario del motor y que la carga gira solidaria - mente con este eje. Este modelo ha sido implementado en Simulink y se muestra en la figura 3.1 en cuya salida se tiene la velocidad angular del secundario del motor; sin carga, la fricción de Coulomb es modelada como $f(n\omega) = C1Sgn(n\omega_k)$.

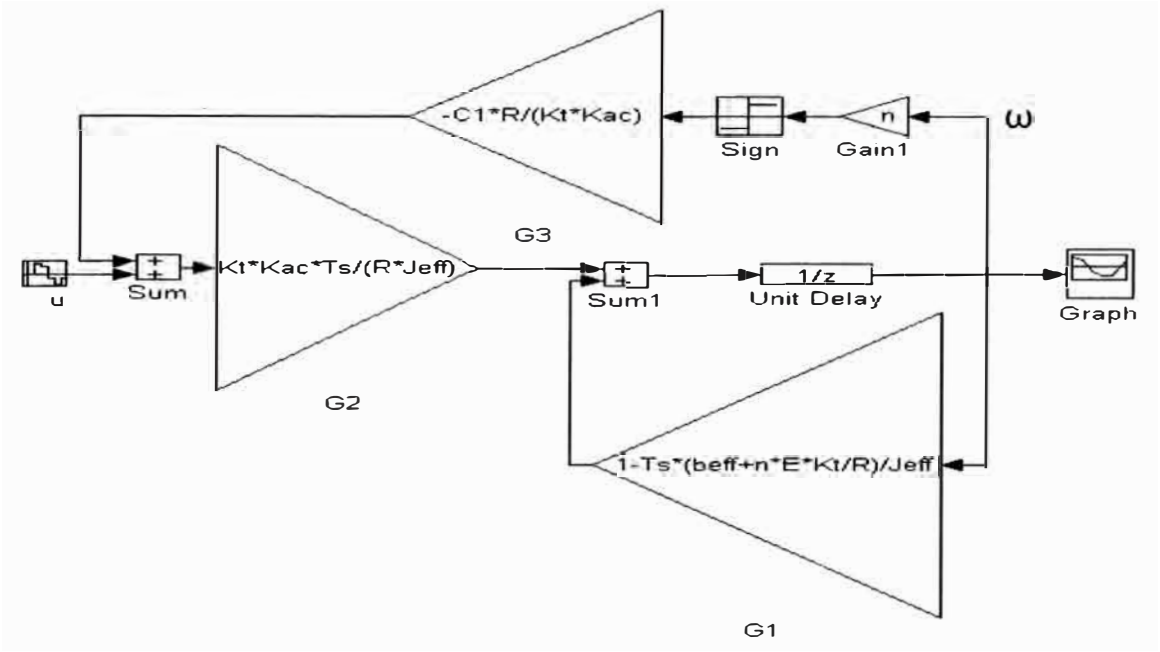

Figura 3.1:Modelo discreto del sistema con salida en la velocidad angular del secundario del motor y sin carga.

En la figura 3.1 se tiene como salida, la velocidad angular ω del eje secundario del motor, para obtener el modelo en que la salida sea la posición angular θ del eje secundario del motor, debemos utilizar un integrador en el dominio discreto dado por el término $\frac{Ts}{1-z^{-1}}$, lo que se tiene en la figura 3.2, adicionalmente en esta figura tenemos el torque T_E, debido a los pesos y que depende del seno de la posición angular.

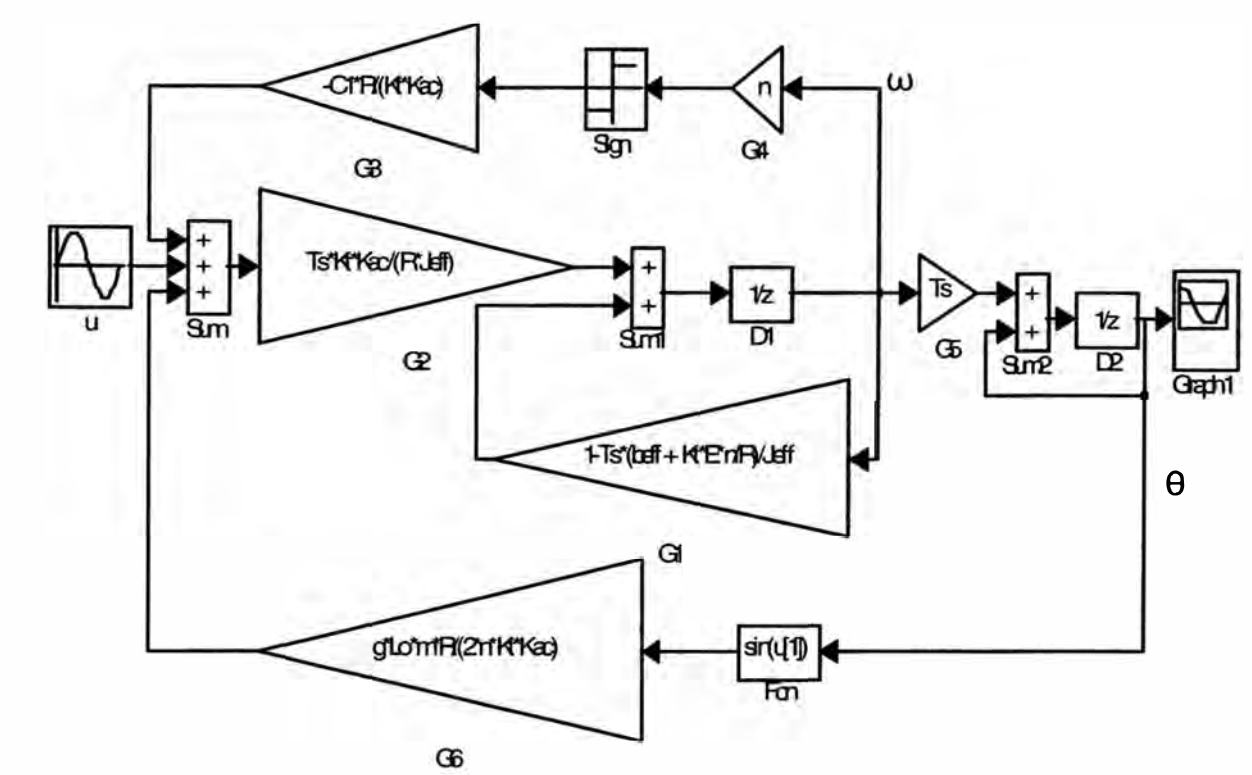

Figura 3.2: Modelo discreto del sistema con salida en la posición angular.

3.2.1 Discretización formal de la planta

Para discretizar la planta no lineal de motor y varilla usamos el software de Matlab. En Matlab para discretizar una ecuación de estado en tiempo continuo podemos utilizar el comando c2dm:

$$[numd, dend] = c2dm(numc, denc, T, 'zoh') \qquad (3.19)$$

La ecuación (3.19) la cual halla el modelo de tiempo discreto a partir del mo - delo en tiempo continuo asumiendo un retenedor de orden cero. Aplicando este comando en nuestro sistema lineal en tiempo continuo dado por la ecuación (3.6), hallamos los polinomios del modelo en tiempo continuo con los siguientes datos:n = 19.741, R = 7.38, Kt = 31.07×10^{-3}, E = 31.0352×10^{-3}, Kac = 14.9, Jeff = 5.63×10^{-5}, beff = 7.05×10^{-5} y Ts = 0.01.

Siendo: M = n*Jeff, B = n*beff y N = 0. Con estos valores de M, B, y N se puede obtener la función de transferencia del motor en continua dada por la ecuación (3.20) siguiente:

$$H(s) = \frac{K}{s(s+a)} \qquad (3.20)$$

Reemplazando los datos dados en las expresiones de K y a que se mues - tran a continuación: $K = \frac{nKacKt}{MR}$ y $a = \frac{BR + n^2Kt}{MR}$. En la ecuación (3.19) te - nemos: $numc = K$ y $denc = [1 \ a \ 0]$. Aplicando el comando c2dm y sabien - do que los coeficientes del sistema en discreto son: a_1 = dend(2), a_2 = dend(2), b_1 = numd(2), b_2 = numd(3). Para el motor obtenemos: a_1 = -1.6245, a_2 = 0.6246, b_1 = 0.0479, b_2 = 0.0410. Con estos parámetros el sistema es lineal, estos parámetros varían ligeramente, como se puede comprobar ali - mentando con datos de entrada y salida en un programa de Matlab espe - cialmente durante el corto tiempo inicial en que la referencia es variable, para mantener la linealidad de la planta del motor con carga en el primer segundo se ajusta con mínimos cuadrados, lo que lleva a anular el error, pues que un error posterior sea mayor a uno precedente, es como reiniciar el proceso, luego con las etapas de estimación y de control se va ajustando poco a poco en cada ciclo de discretización, hasta que el conjunto llega a la estabilidad, en el control de posición en este caso para una masa de la varilla de 0.06377 Kg. y una longitud de 0.776 m; los parámetros de la planta a_1, a_2, b_1 y b_2 que dependen de la frecuencia de muestreo Fs, los consideramos como constantes y son ajustados por el programa.

Las matrices G y H que identifican a la planta se pueden también obtener fácilmente con Matlab, aplicando la orden:

$$[G,H] = c2d(A,B,Ts] \tag{3.21}$$

Donde Ts denota el periodo de muestreo. Considerando que nuestro sistema es lineal e invariante en el tiempo, el sistema puede representarse como:

$$\begin{aligned} X(k+1) &= GX(k) + HU(k) \\ Y(k) &= CX(k) \end{aligned} \tag{3.22}$$

La función de transferencia discreta de la planta es:

$$\frac{Y(z)}{u(z)} = Gp(z) \tag{3.23}$$

La cual puede expresarse de la forma:

$$Gp(z) = \frac{b_1 z + b_2}{z^2 + a_1 z + a_2} \tag{3.24}$$

Luego, podemos obtener la forma canónica controlable:

$$G = \begin{bmatrix} 0 & 1 \\ -a_2 & -a_1 \end{bmatrix}; \qquad H = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \qquad C = \begin{bmatrix} b_2 & b_1 \end{bmatrix}$$

Entonces, la ecuación en tiempo discreto es:

$$X(k+1) = \begin{bmatrix} 0 & 1 \\ -a_2 & -a_1 \end{bmatrix} X(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U(k)$$

$$Y(k) = \begin{bmatrix} b_2 & b_1 \end{bmatrix} X(k) \tag{3.25}$$

La discretización de la planta es posible, la señal de control se mantiene en el puerto del DSP por el tiempo de discretización establecido por el contador del temporizador del DSP y trabajando con la frecuencia de operación de 40 Megahertz, la salida de posición angular y sigue a la señal de entrada u, la posición angular sensada de salida se obtiene con ocho bits pudiéndose leer en otro puerto del DSP con la instrucción de lectura.

Discretización de la planta usando Simulink

Para obtener un modelo discreto a partir de uno continuo en Simulink, primero hay que obtener el modelo continuo y luego agregamos bloques de zero-order hold a la entrada y salida de la planta, teniendo en cuenta el tiempo Ts de discretización. Para obtener el modelo continuo, de las ecuaciones diferenciales de la planta despejamos las derivadas de mayor orden, en el presente caso la derivada de la velocidad angular del eje secundario del motor y la derivada de la corriente, procedemos utilizando los bloques de integración como medio de orientación, a continuación ayudados por el álgebra, dibujamos el modelo. Para presentar el modelo de la planta, demostraremos usando Simulink, que la respuesta de la salida de posición es la misma si se considera un sistema con la pequeña variación de la corriente del motor, que tiene en cuenta la inductancia de armadura, en la figura 3.3, cuya salida de posición se tiene en la figura 3.4, con el otro sistema estudiado, que considera a la corriente de armadura constante, en la figura 3.5 y cuya salida de posición se ve en la figura 3.6. Alimentando con una onda seno de 0.5 voltios de amplitud y una frecuencia angular de 3 Radianes/seg ambos modelos individualmente. En la figura 3.5, para obtener el modelo se despejó de las ecuaciones diferenciales de la planta la corriente de armadura(pues consideramos que la derivada de la corriente de armadura es nula) y la derivada de la velocidad angular. En ambos modelos mencionados se consideró la ganancia del generador PWM (Kac) independiente del modelo y necesaria para activar cualquier modelo que usemos. Las figuras 3.4 y 3.6 muestran la misma respuesta de posición

angular, lo que comprueba lo establecido en la sección 2.2, indiferentemente podemos usar cualquiera de los modelos de las figuras 3.3 ó 3.5 para simular el sistema. Finalmente el modelo discretizado se muestra en la figura 3.7, cada bloque de zero-order es un sample and hold nombrados como Hzoh(z) que debe indicar el tiempo de discretización Ts, recibe como entra - da una muestra señal continua y la convierte en su equivalente discreto en su salida. En la figura 3.7, el modelo que se encuentra entre los dos bloques de zero-order puede ser agrupado por Simulink en un bloque con entrada (IN) y salida (OUT) para simplificar y reducir el tamaño de los modelos sobre todo cuando se emplea controlador, tal como se tiene para el esquema de la planta con controlador, en sección 6.2 en la figura 6.1.

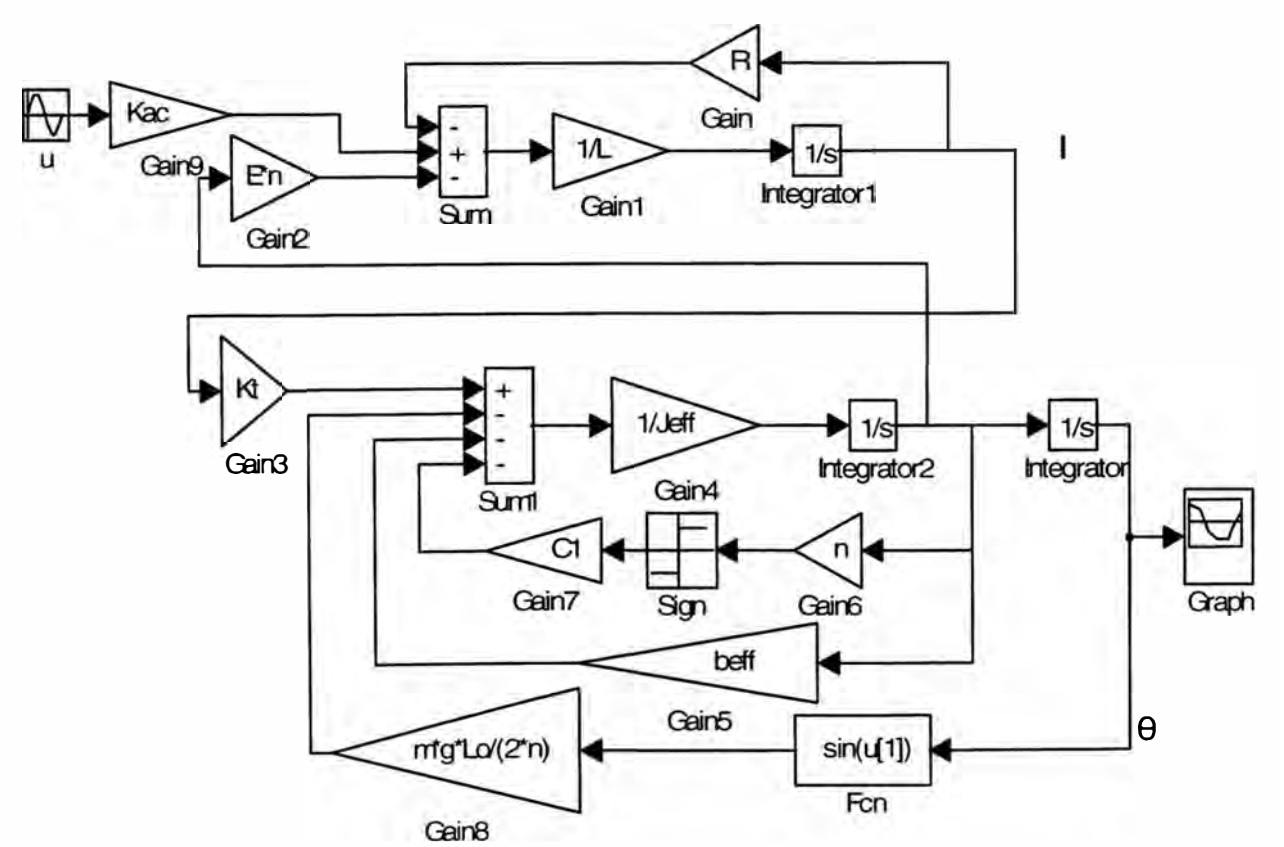

Figura 3.3: Modelo del motor considerando la pequeña variación de la co - rriente de armadura.

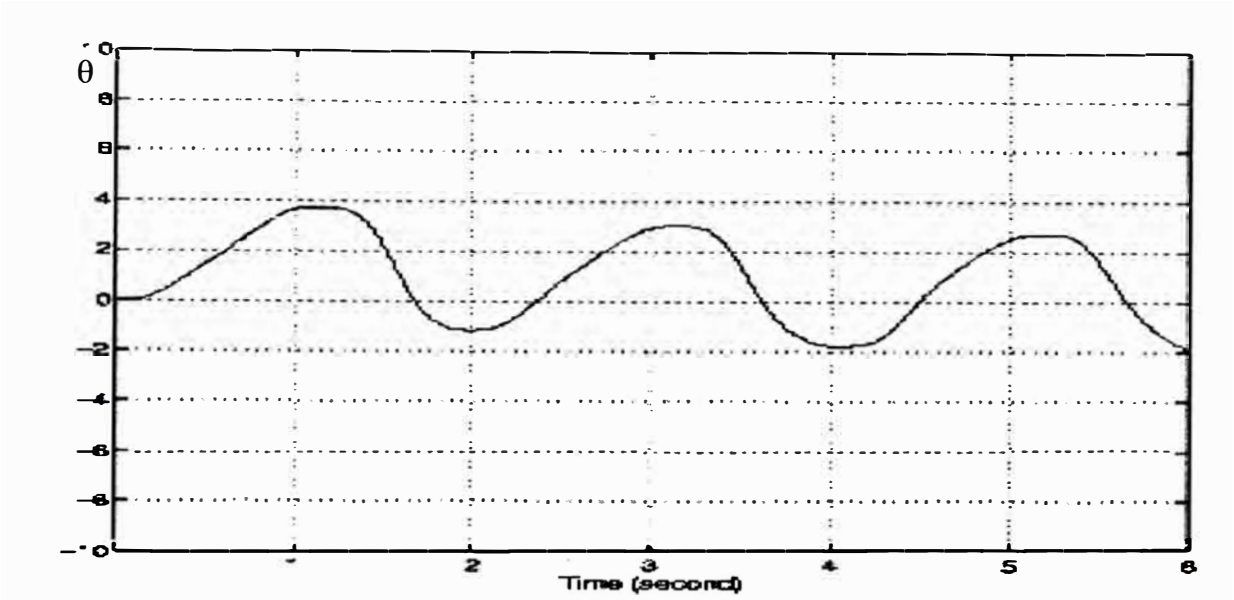

Figura 3.4: Respuesta de posición del modelo de la planta considerando la variación de la corriente de armadura.

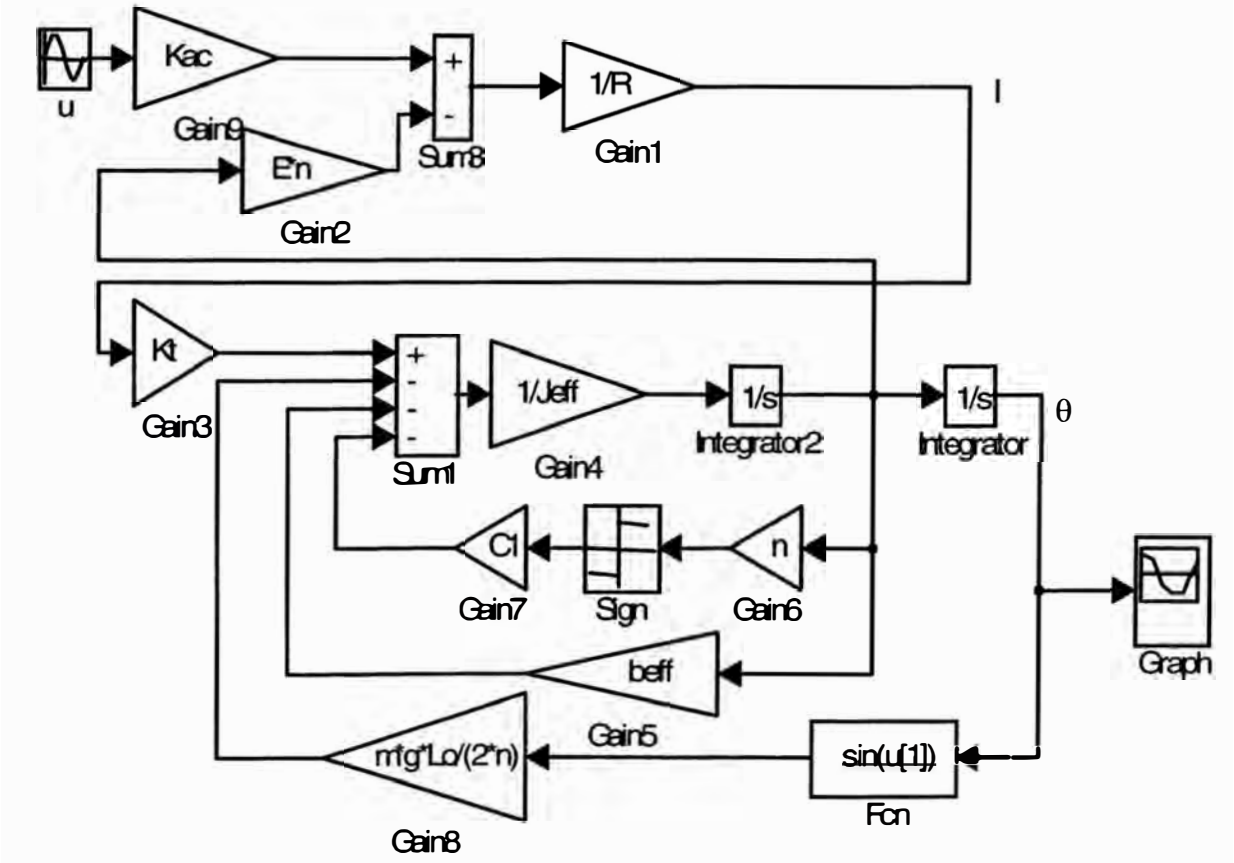

Figura 3.5: Modelo del motor, considerando que la corriente de armadura permanece constante, con la excitación de voltaje que la figura 3.3, con el objeto de comparar sus salidas.

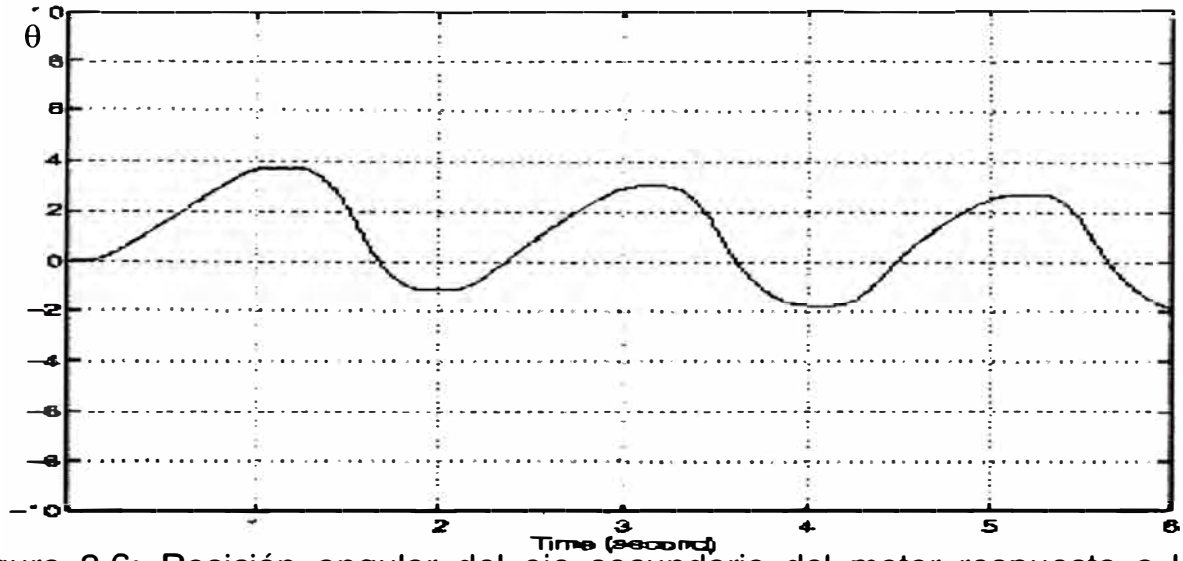

Figura 3.6: Posición angular del eje secundario del motor respuesta a la misma excitación de la figura 3.3, considerando que la corriente de armadura permanece constante por que estamos tratando con un motor de corriente continua y tomando la misma salida que el modelo de la figura 3.3.

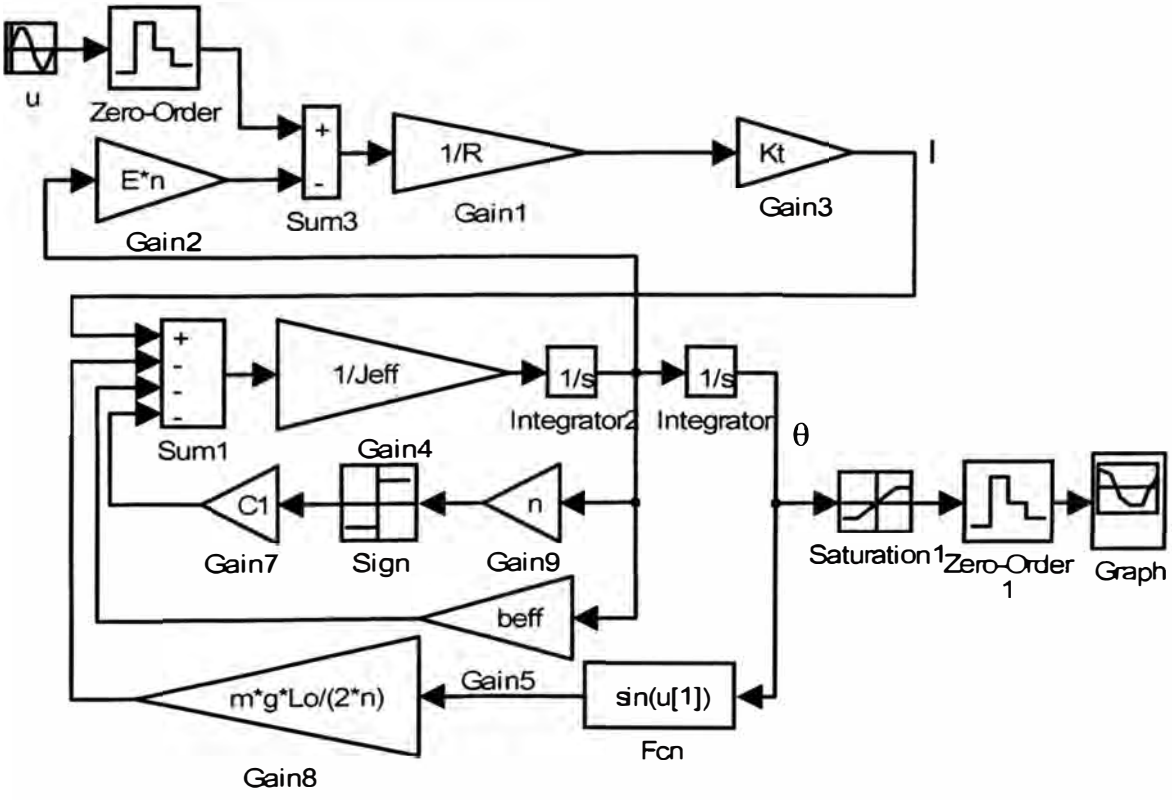

Figura 3.7: Modelo discretizado en Simulink del motor con eje secundario.

3.2.2 Estabilidad de la planta usando comandos de Matlab

Trabajando en tiempo discreto para saber sobre la estabilidad de la planta motor y varilla se hicieron tres alcances:

1. La controlabilidad de la planta.
2. La observabilidad de la planta.
3. La estabilidad en el dominio discreto.

Controlabilidad de la planta

Se realizó el estudio basándonos en los parámetros constantes y considerando que sé esta trabajando en el eje secundario del motor:

Controlabilidad de la planta de motor y varilla

Se utiliza el comando Ctrb(A,B), donde: $A = \begin{bmatrix} 0 & 1 \\ -0.6246 & 1.6246 \end{bmatrix}$; $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$; $C = [0.041 \quad 0.0479]$. Con la aplicación del comando mencionado, C_0=Ctrb(A,B). Se obtuvo: $C_0 = \begin{bmatrix} 0 & 1.0 \\ 1.0 & 1.6246 \end{bmatrix}$. Cuyo rango de esta matriz es 2, ósea: rank(C_0) = 2. Por lo que el sistema es controlable.

Observabilidad de la planta de motor y varilla

Se realizó el estudio de Observabilidad de la planta de motor y varilla. Se utiliza el comando Obsv(A, B), donde: A, B y C son las mismas matrices de estado que para el caso del estudio de controlabilidad. Con la aplicación del comando mencionado, Ob=obsv(A,C). Se obtuvo: $Ob = \begin{bmatrix} 0.0410 & 0.0479 \\ 0.0299 & 0.1188 \end{bmatrix}$.

El rango de esta matriz es rank(Ob)=2. Con lo que la planta es observable.

La estabilidad en el dominio discreto

La estabilidad se analizó mediante el comando Dnyquist.m.

Estabilidad en el dominio discreto del motor y varilla

Mediante el comando de Dnyquist.m se obtiene la respuesta en fre - cuencia de la planta, como sabemos si la gráfica de la respuesta en fre – cuencia está incluida dentro del circulo unitario el sistema es estable, para evaluar se requiere los coeficientes en tiempo discreto de la planta y el periodo de discretización que es 0.01 segundos, con el comado C2DM de Matlab, obtenemos a partir de los coeficientes en tiempo continuo los coeficientes en tiempo discreto. Si tenemos num y den los polinomios de la función de tranferencia en tiempo continuo con la ecuación (3.26) siguiente:

$$[numd,\ dend] = C2DM(num,\ den,\ 0.01,'zoh') \qquad (3.26)$$

Obtenemos los coeficientes de los polinomios de la función de transferencia de la planta en tiempo discreto, utilizando estos últimos en el comando ex - presado por la ecuación siguiente: DNYQUIST(numd, dend, 0.01), Matlab muestra los gráficos de respuesta en frecuencia en el plano complejo. Con la aplicación de los coeficientes de la función de transferencia del sistema en tiempo continuo siguientes: num = [0.0479 0.0410] y den = [1.00 -1.6246 0.6246] en la ecuación 3.26, obtenemos los coeficientes en el tiempo discreto correspondiente y con el comando Dnyquist se tiene la gráfica 3.8 de respuesta en frecuencia. Por el criterio de Nyquist como la gráfica está incluida en el circulo unitario del plano complejo, el sistema es estable.

3.3 Señales para muestreo de la información

En los controladores de acuerdo al número de instrucciones que son requeridas se puede hallar el mínimo periodo de muestreo. Así por ejemplo, para 12000 instrucciones (i) requeridas y una frecuencia de operación de 40 Megahertz (f_0) del procesador, el mínimo periodo de muestreo (mp) viene dado por la fórmula (3.27) siguiente:

$$mp = \frac{2i}{f_0} \tag{3.27}$$

En el ejemplo se obtiene 600 microsegundos, lo que quiere decir que podemos utilizar una duración mayor ó igual a la calculada. En nuestro sis - tema con DSP la frecuencia de operación es obtenida utilizando el dispositi - vo PLL que es un registro de 24 bits que se programa para tal fin, el cristal que trabaja con el DSP es de 4 Megahertz lo que corresponde a un periodo de 0.25 microsegundos, la salida en el pin TIO se tiene en la figura 3.9, con esta frecuencia de operación, la máxima frecuencia de trabajo corresponde a 2 Megahertz por el teorema del muestreo.

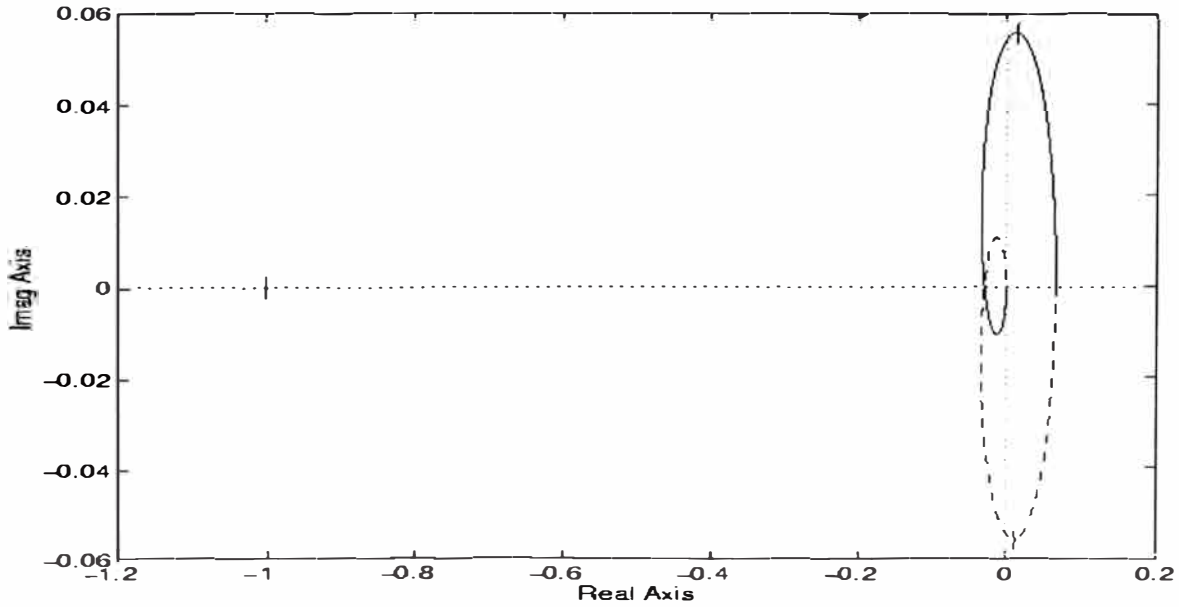

Figura 3.8: Gráfica de Estabilidad en el dominio discreto del sistema.

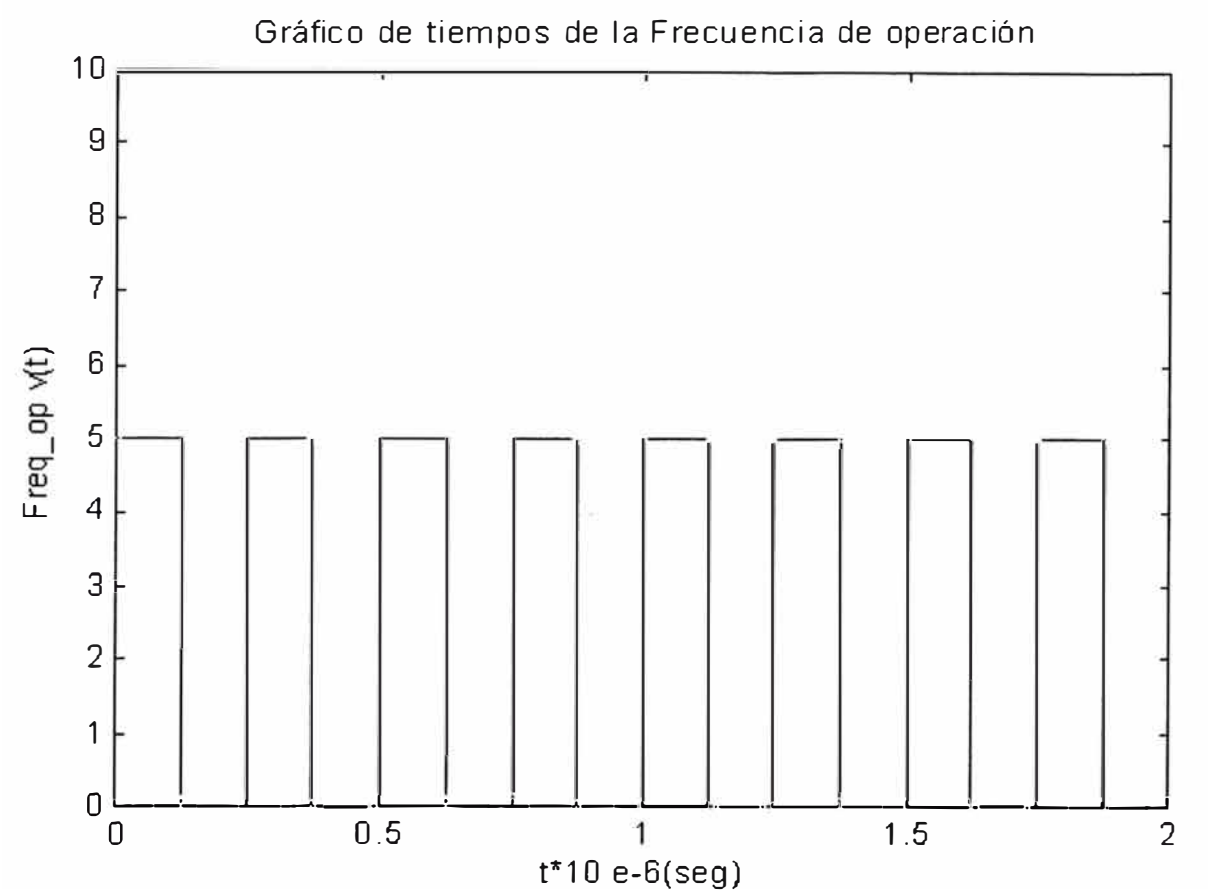

Figura 3.9: Gráfico de la frecuencia de operación brindada por el cristal del procesador DSP.

Por medio del PLL se programó para que la frecuencia de operación sea 40 Megahertz, que corresponde a un periodo de 25 nanosegundos, lo que se muestra en la figura 3.10, en donde también por el teorema del muestreo el periodo mínimo de la mayor frecuencia de trabajo es de 50 na - nosegundos. Como hemos simplificado nuestro programa por las considera - ciones expuestas y dado que desde el programa en lenguaje C se ajustó a un tiempo de discretización de 50 milisegundos, el tiempo de discretización usado en el programa en lenguaje ensamblador que se utilizó fue el tiempo de discretización de 50 milisegundos, se logra este tiempo por medio del temporizador del DSP donde se coloca un número adecuado en el contador

de dicho temporizador para que cada 50 milisegundos se renueve el valor de la señal de control u, el programa tiene todo ese tiempo para realizar las operaciones necesarias, el número dicho se obtiene por observación en el osciloscopio al comprobar la duración de los diferentes valores de u ó también del tiempo total de la excursión que es de 20 segundos.

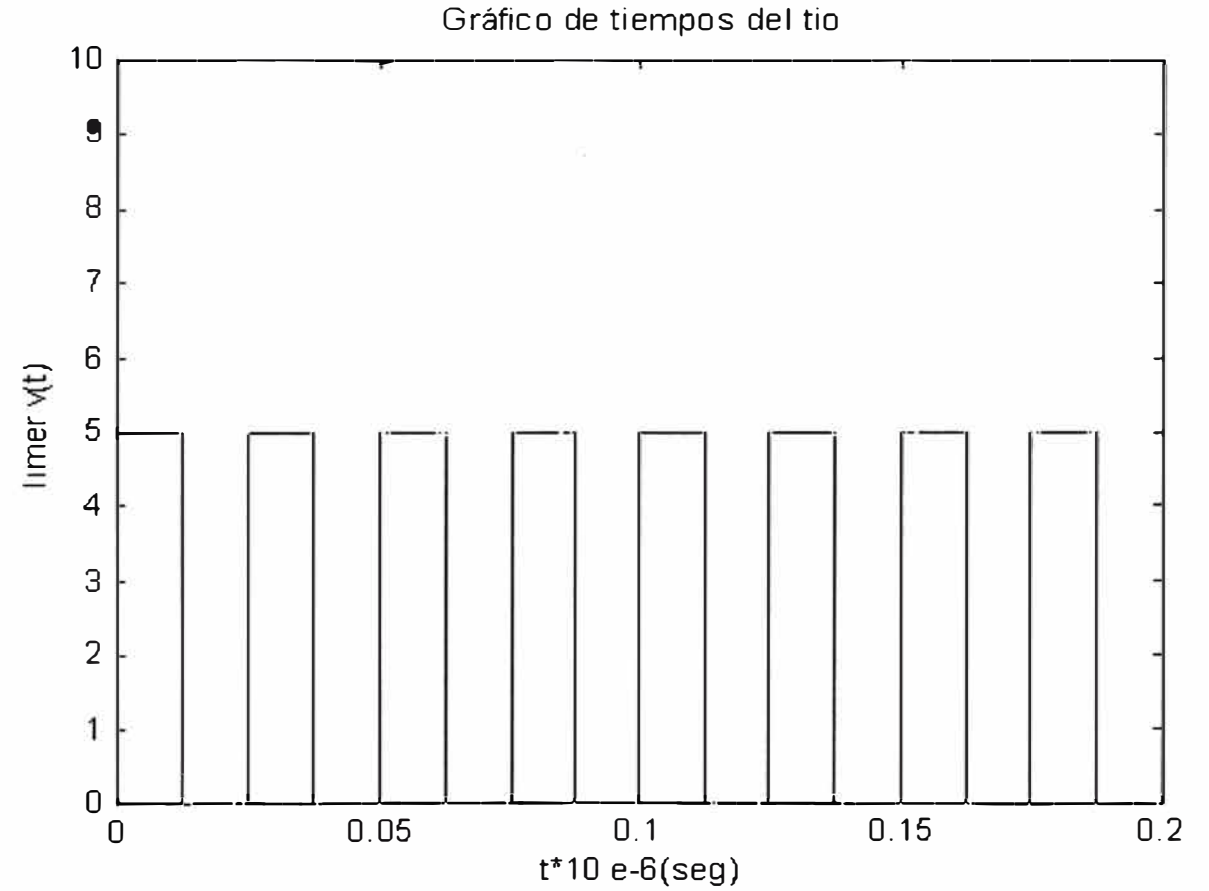

Figura 3.10: Gráfico de la frecuencia de operación ajustado por el PLL del DSP, esta señal es observada en el pin TIO.

CAPÍTULO IV
TEORÍA Y ALGORITMOS

En este capítulo se tratará la teoría utilizada para explicar el diagra - ma de bloques del control de posición CAA (controlador auto-ajustable) re - presentado en la figura 4.1, donde observamos que este controlador CAA combina en su diseño el método de estimación de parámetros con el modelo de representación de proceso, y una ley de control (el controlador auto - ajustable). Se asume que es posible modelar el proceso no lineal como el de un modelo de proceso lineal que se vio en el capítulo III. El método de identificación de parámetros utilizado es el de mínimos cuadrados recursivo (RLS), que al ser probado es llamado probado RLS (IRLS). La ley de control utilizada es un controlador de realimentación de espacio, estados cuadráticos, lineal y proporcional e integral (PI-LQSSFC). El objetivo del control CAA es elegir una función con la fuerza capaz de reducir al mínimo la diferencia entre la salida y la referencia. El CAA representado en la figura 4.1 opera como sigue: Después de cada intervalo de tiempo de muestreo, puede ser actualizado el vector de estimación de parámetros $\hat{\vec{\theta}}$ por los datos de entrada U y salida Y de información. Los elementos del vector de estimación de parámetros pueden ser usados para recuperar el modelo de proceso lineal, que permitirá la estimación del modelo de proceso del vector de

estados $\vec{x}$ (usando un filtro de Kalman), y el valor de referencia U^0 de la actual ley de control. Tales estimaciones serán utilizadas para computar la ley residual u de control y renovar la ley de control real en la relación $U = U^0+u$. También se explicará los algoritmos de control que fueron adecuados en lenguaje de programación en C de forma gradual a como iba generando - se el programa en lenguaje ensamblador y dado que la respuesta en tiempo real fuese factible se pudo ir incrementando etapas de algoritmo en el lenguaje ensamblador para tener la respuesta del DSP a estos algoritmos que ya fueron comprobados y que fueron posibles de implementar en tiempo real.

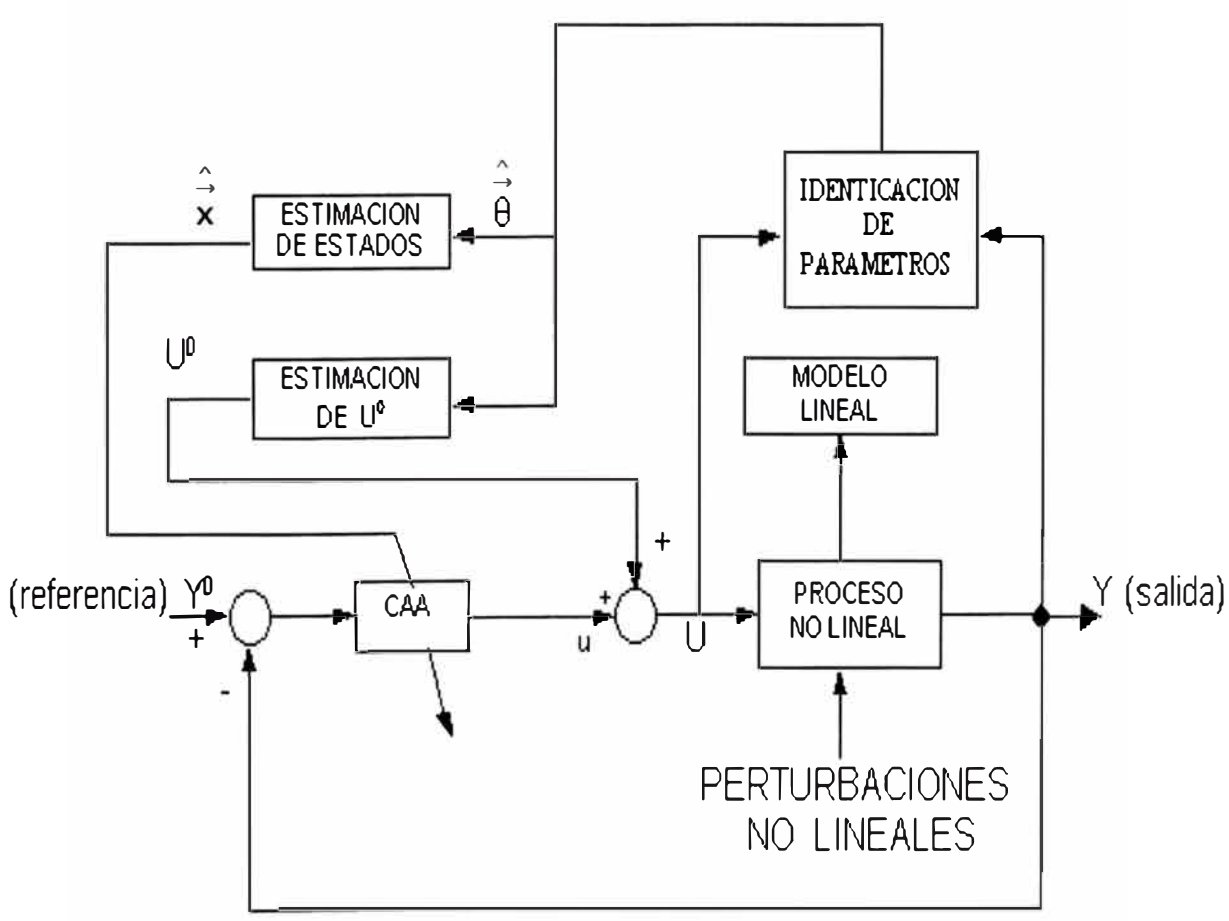

Figura 4.1: Configuración de un sistema de control auto-ajustable(CAA).

Un aspecto importante y que permite tener que la posición sea igual a la referencia desde los primeros instantes de iniciado el proceso es el de re - gresión lineal, en la referencia [13] es explicado como el método de regre - sión lineal predice la variable controlada de salida como función de una ó más variables independientes, además se tiene la expresión polinómica de la salida relacionando el método de mínimos cuadrados aplicado a las des - viaciones de dichos coeficientes, siendo este polinomio una manera válida de predecir el comportamiento del modelo, la labor hecha puede verse en la figura 4.2 en la cual la curva de la posición es la que se inicia en cero la otra curva de trazo continuo es la referencia ambas señales corresponden a las respuestas del programa de prueba en lenguaje C++ para tiempo real.

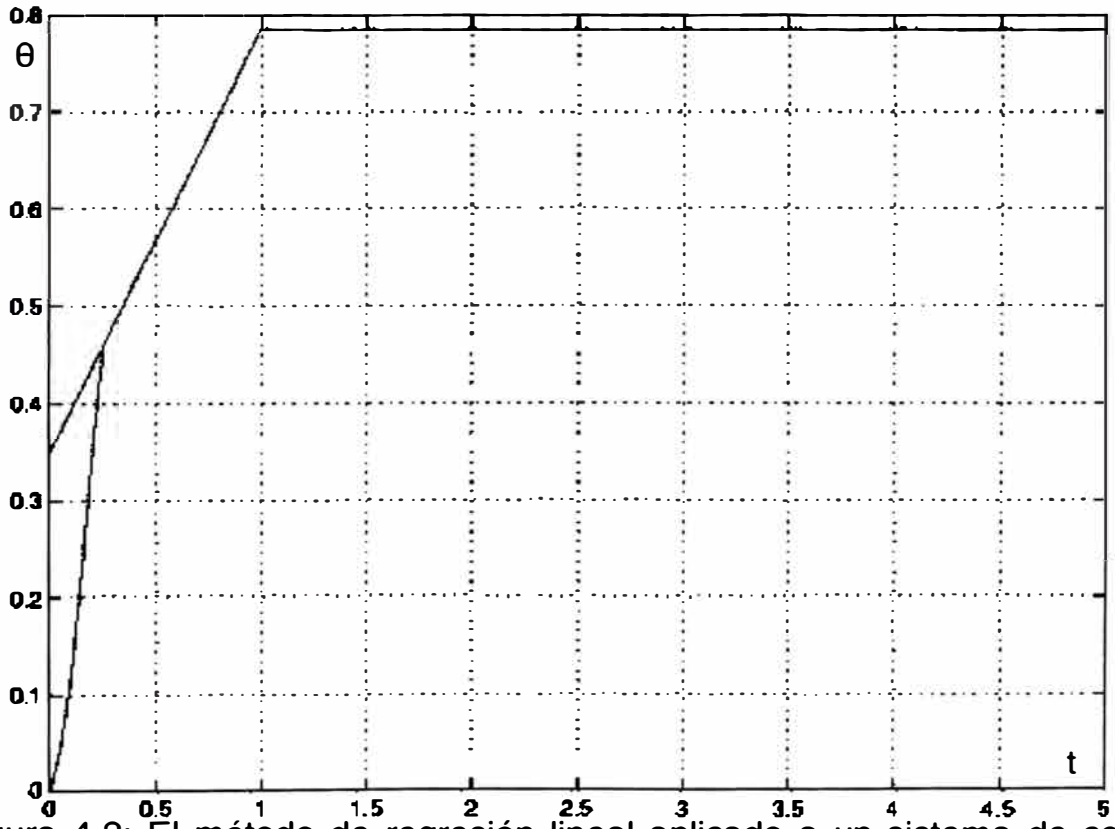

Figura 4.2: El método de regresión lineal aplicado a un sistema de control CCA de posición.

4.1 Bases teóricas

El desarrollo que se presenta tiene el siguiente orden:

1. Procedimientos de estimación.
2. La ley de control.
3. Procedimiento de diseño del CAA.

4.1.1 Procedimientos de estimación

En esta sección presentamos procedimientos para estimar en línea modelos de parámetros y procesos de estados.

El método IRLS

Para estimación de los procesos de parámetros, la descripción del espacio de estados viene dado por la ecuación (4.1) tiene que reinicializar bajo la siguiente forma:

$$y(k) = \vec{\Psi}^T(k)\,\vec{\theta} \tag{4.1}$$

Donde la información del vector de mediciones $\vec{\Psi}^T$ contiene valores presen - tes y pasados del proceso de entrada u y del proceso de salida y, el vector de estimación $\vec{\theta}$ contiene los parámetros a ser estimados. El polinomio tiene la forma dada en la ecuación (3.5). Con las siguientes relaciones:

$y=Y-Y^0$ y $u=U-U^0$ en (3.5), obtenemos la siguiente ecuación:

$$A(q^{-1})Y(k) = B(q^{-1})U(k) + C(q^{-1}) \tag{4.2}$$

$$C(q^{-1}) = A(q^{-1})Y^0(k) - B(q^{-1})U^0(k) \tag{4.3}$$

Por consiguiente, la ecuación (4.1) toma la siguiente forma:

$$Y(k) = \begin{bmatrix} \vec{\Psi} & 1 \end{bmatrix} \begin{bmatrix} \vec{\theta} \\ C \end{bmatrix} \tag{4.4}$$

En la ecuación (4.4), el orden de la información resultante y vector de parámetros tiene que incrementarse en uno. La nueva información del vector $\begin{bmatrix} \vec{\Psi}^T & 1 \end{bmatrix}$ contiene valores pasados y presentes del proceso actual entrada U y salida Y, además del nuevo parámetro C a ser estimado en línea. Para q=1, ecuación (4.4) representa las condiciones de estado estacionario. Para la condición dada, el valor de referencia $\vec{U}^0$ toma la siguiente forma:

$$U^0(k) = B^{-1}(1)[A(1)\vec{Y}^0(k) - C(k)] \qquad (4.5)$$

Los algoritmos RLS presentan un número potencial de problemas que podrían afectar el diseño del CAA. En este alcance se emplea un método comprobado, en nuestro caso dicha comprobación se realizó en tiempo real mediante un programa en C++ en una microcomputadora Pentium cum - pliendo requisitos fraccionarios del DSP, además mediante programa de software de Matlab, en forma semejante al programa en C++, en que utilizando resultados de tiempo real de entrada y salida del proceso se puede observar como varían los parámetros de la planta hasta llegar a la estabilidad.

Estimación de estados

Podemos obtener de la ecuación (3.4) el siguiente proceso nominal de perturbación:

$$\vec{x}(k+1) = A\vec{x}(k) + Bu(k) + \vec{\upsilon}(k)$$

$$Y(k) = C\vec{x}(k) + \omega(k) \qquad (4.6)$$

Durante el proceso de estimación, el vector de parámetros θ se con - vierte en el vector de parámetros estimados $\hat{\vec{\theta}}$. Utilizando dichas estima - ciones, podemos recuperar los elementos de las matrices estimadas A(k) y $\hat{B}(k)$ en orden a obtener la versión de estimación de la ecuación (3.4) como sigue:

$$\hat{\vec{x}}^{+}(k) = \hat{A}(k)\hat{\vec{x}}^{-}(k) + \hat{B}(k)u(k) \qquad (4.7)$$

$$y(k) = C\hat{\vec{x}}^{-}(k) \qquad (4.8)$$

Donde las estimaciones de estados $\hat{\vec{x}}^{-}$ y $\hat{\vec{x}}^{+}$ pueden ser obtenidas usando un filtro de Kalman con observación de estados:

$$\hat{\vec{x}}^{+}(k) = \hat{\vec{x}}^{-}(k) + F(k)[y(k) - C\hat{\vec{x}}^{-}(k)] \qquad (4.9)$$

Con nuevos estados:

$$\hat{\vec{x}}^{-}(k+1) = \hat{A}(k)\hat{\vec{x}}^{-}(k) + \hat{B}(k)u(k) + \hat{A}(k)F(k)[\vec{y}(k) - C\hat{\vec{x}}^{-}(k)] \qquad (4.10)$$

En las ecuaciones (4.9) y (4.10), la matriz de ganancia F(k) está dada por:

$$F(k) = \quad (k)C^{T}[C \quad (k)C^{T} + \quad (k)]^{-1} \qquad (4.11)$$

Donde: $\Sigma(k)=\Sigma^{T}(k)$ es la solución única definida positiva de la ecuación aso - ciada a la matriz de Riccati discreta en la página siguiente:

$$(k) = \hat{A}^{T}(k)\ (k)\ (k)\hat{A}^{T}(k) + \ (k) - \hat{A}F(k)C(k)\ (k)\hat{A}^{T}(k) \qquad (4.12)$$

Donde: Θ(k) y Ξ(k) son las covarianzas definidas positivas de las perturba - ciones υ y ω.

4.1.2 La ley de control

El proporcional LQSSFC es una matriz de ganancia K_x, tal que la ley de control $u=-K_x x(k)$, minimizando la siguiente función de costo:

$$I_{LQSSFC} = \sum_{k=0}^{\infty}[\vec{x}^{T}(k)Q\vec{x}(k) + u^{2}(k)R] \qquad (4.13)$$

Sujeta a la siguiente ecuación:

$$\vec{x}(k+1) = A\vec{x}(k) + Bu(k)$$

Donde la matriz $Q=Q^{T} \geq 0$ es semidefinida positiva y la matriz $R > 0$ es definida positiva. La ganancia K_x está dada por:

$$K_x = (R + B^{T}SB)^{-1}B^{T}SA \qquad (4.14)$$

Donde S es la solución única definida positiva de la siguiente ecuación matricial asociada de Riccati discreta:

$$0 = S - A^{T}SA + A^{T}SBK_x \qquad (4.15)$$

}

Para poder comprobar la perfomance del proporcional LQSFC, es posible añadir una acción integral al controlador. Escogiendo la variable z(k) como la integral (sumatoria) del sistema de error $Y^{0}(i)$-Y(i) como sigue:

$$z(k) = \sum_{i=0}^{k-1}[Y^{0}(i) - Y(i)] = \sum_{i=0}^{k-1}[-y(i)]$$

$$z(k+1) = \sum_{i=0}^{k}[-y(i)]$$

Por consiguiente:

$$z(k+1) = z(k) - y(k) = z(k) - C\,x(k) \qquad (4.16)$$

Entonces, la representación de espacio de estados aumentada toma la siguiente forma:

$$\vec{x}^{a}(k+1) = A^{a}\vec{x}^{a}(k) + B^{a}u(k) \tag{4.17}$$

$$y(k) = C^{a}\vec{x}^{a}(k) \tag{4.18}$$

En las ecuaciones (4.17) y (4.18) el superíndice a es la denotación para ma - triz aumentada. El vector $\vec{x}^{a}$ y las matrices A^{a}, B^{a}, y C^{a} toman la siguiente forma: $\vec{x}^{a} = \begin{bmatrix} \vec{x}(k) \\ z(k) \end{bmatrix}$, $A^{a} = \begin{bmatrix} A & 0 \\ -C & I \end{bmatrix}$, $B^{a} = \begin{bmatrix} B \\ 0 \end{bmatrix}$, $C^{a} = [C \quad 0]$.

Nuestro problema es encontrar la matriz de ganancia K_{xa} tal que la ley de control minimice la siguiente función de costo:

$$I^{a}_{LQSSFC} = \sum_{k=0}^{\infty}[(\vec{x}^{a})^{T}(k)Q^{a}\vec{x}^{a} + u^{2}(k)R] \tag{4.19}$$

Donde la matriz $Q^{a} = [Q^{a}]^{T}$ es semidefinida positiva. La ganancia K_{xa} viene dada por:

$$K_{xa} = \{R + [B^{a}]^{T}S^{a}B^{a}\}^{-1}[B^{a}]^{T}S^{a}A^{a} \tag{4.20}$$

En la ecuación (4.20), S^{a} es la solución única definida positiva de la si - guiente ecuación matricial asociada de Riccati discreta:

$$0 = S^{a} - [A^{a}]^{T}S^{a}A^{a} + [A^{a}]^{T}S^{a}B^{a}K_{xa} \tag{4.21}$$

Por consiguiente, la ley de control del CAA viene dada por:

$$u(k) = -K_{xa}\vec{x}^{a}(k) \tag{4.22}$$

Los parámetros de orientación de la perfomance son R y Q^{a}, que son ajustados para una mejor respuesta del sistema.

4.1.3 Procedimiento del diseño del CAA

El procedimiento de diseño para SCAA mostrado en la figura 4.1 es como sigue:

1. Utilizar toda la información acerca del proceso no lineal, determi - nando su modelo de proceso lineal.

2. Implementar el método IRLS por estimación del proceso de pará - metros, e implementando el desarrollo del procedimiento de estimar el pro - ceso del modelo de estados.

3. Implementar la ley de control residual u(k) dada por la ecuación (4.22), computar el vector de referencia $U^0(k)$ usando la ecuación (4.5), y actualizar la ley de control actual usando la relación $U(k) = u(k)+U^0$.

4. Finalmente determinar los parámetros de orientación de la perfo - mance que son R y Q^a.

4.2 Diagramas de flujo del programa implementado

Se realizaron los siguientes diagramas de flujo que explican el pro - grama implementado en el DSP, los cuales se enumeran a continuación, que fueron:

1. Diagrama de flujo del programa de espera a interrupción.

2. Diagrama de flujo del programa a implementar.

3. Diagrama de flujo de mejoramiento de la señal de control.

4. Diagrama de flujo del cálculo de la posición angular de salida.

4.2.1 Diagrama de flujo del programa de espera a interrupción.

En este programa se siguen los lineamientos generales para destacar la importancia de la subrutina, el bloque de inicio incluye todos los registros, programas que se incluyen y posiciones de memoria utilizadas, al principio evitamos el salto a la subrutina x1 y entramos por la primera instrucción posterior a este salto a subrutina y a la cual retornará el programa mediante orden RTS después de cada orden de espera a interrupción y luego se emplea menos de 0.05 segundos para completar las operaciones necesarias para tener la nueva señal de control a enviar si es que se ha cumplido el tiempo de discretización de 0.05 segundos, el sistema se interrumpe, luego, se procede al salto a la subrutina x1, la cual envía el nuevo valor de la señal de control y offset hacia el generador PWM, luego a la orden de espera a interrupción si es que no se ha cumplido con el tiempo de excursión de 20 segundos, posteriormente se retornará al programa a la primera instrucción posterior al salto a subrutina x1 para repetir el proceso, el diagrama de flujo se tiene en la figura 4.3.

4.2.2 Diagrama de flujo del programa a implementar.

Este programa tiene la implementación del control de posición con el sistema CAA en la figura 4.4.

4.2.3 Diagrama de mejoramiento de la señal de control.

Este programa tiene la implementación del mejoramiento de la res - puesta del control de posición con el sistema CAA en la figura 4.5, con la ayuda de datos en tiempo real, de las salidas de programa en C, que son usados como condiciones límites en el programa ensamblador para así ajustar la respuesta y mejorar nuestro modelo cuya salida se pudo suavizar discretizando y teniendo en cuenta el efecto de escalera (tanto la entrada como la salida del sistema deben avanzar en un sentido, subiendo ó bajando, tomando en cuenta valores de piso y de techo), con esto se logra los objetivos de la técnica de mínimos cuadrados, la convergencia a cero del error, valores constantes de las señales durante casi todo el tiempo que dura el proceso, evitando en lo posible se inicie un nuevo proceso, es lo que se tiene en el diagrama de la figura 4.5.

4.2.4 Diagrama de flujo del cálculo de la posición angular de salida.

Del contador de pulsos del sensor se obtiene el incremento de la po - sición angular del motor, que es leída en un puerto paralelo del DSP, se multiplica por $2\pi/512*19.742$, para tener el incremento de la posición angular de salida, considerando la reducción, agregando este último incremento a la posición angular de salida guardada en el instante previo de discretización, luego se guarda la última adición en alguna posición de la memoria de datos, como la nueva posición angular de salida, para luego ser restada de la refe - rencia, lo que es el error, que es lo que, debemos hacer converger a cero, lo que se tiene en la figura 4.6.

Diagrama de flujo del programa de espera a interrupción

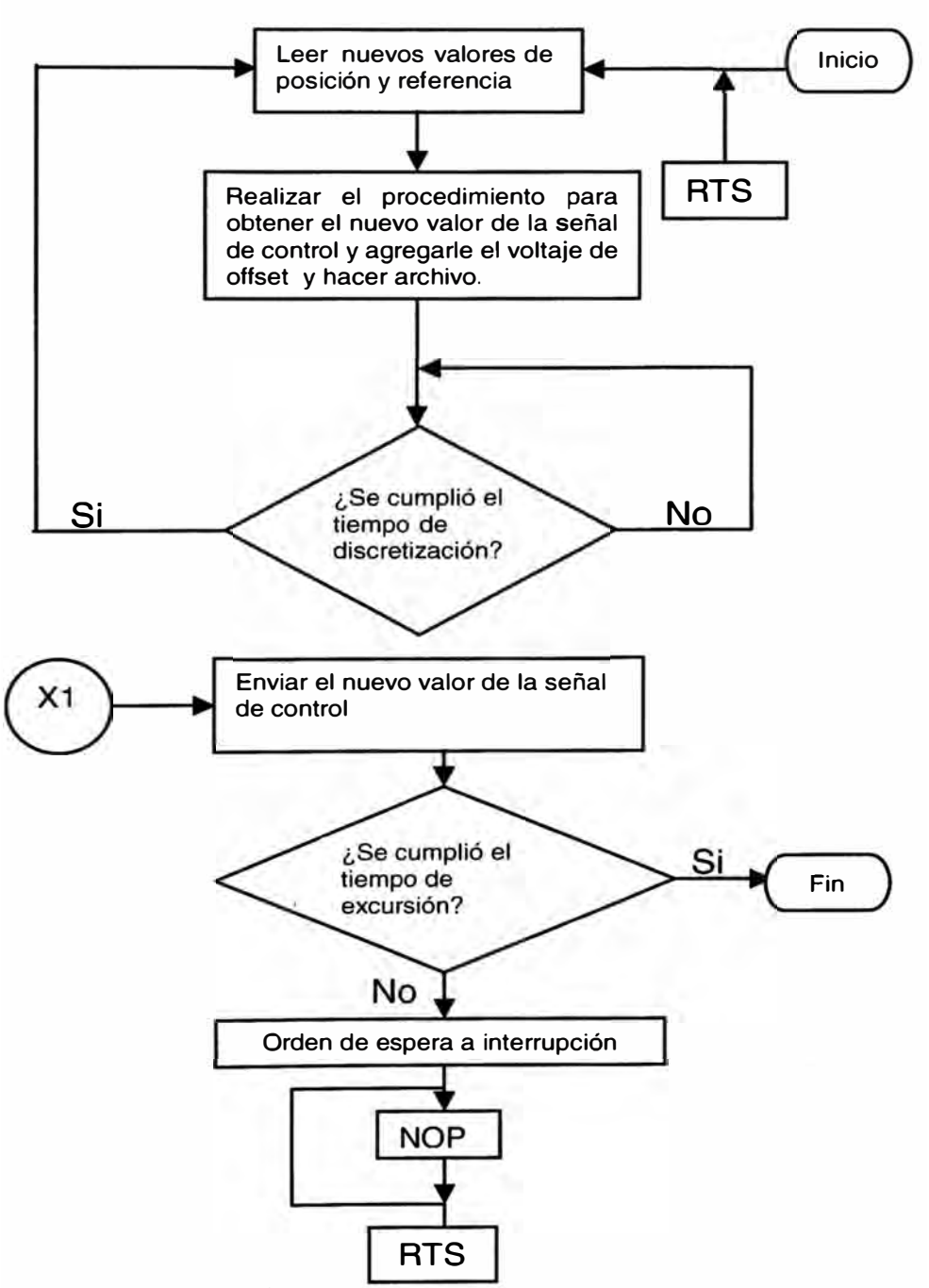

Figura 4.3: Diagrama de flujo del programa de espera a interrupción.

Diagrama de flujo del programa a implementar

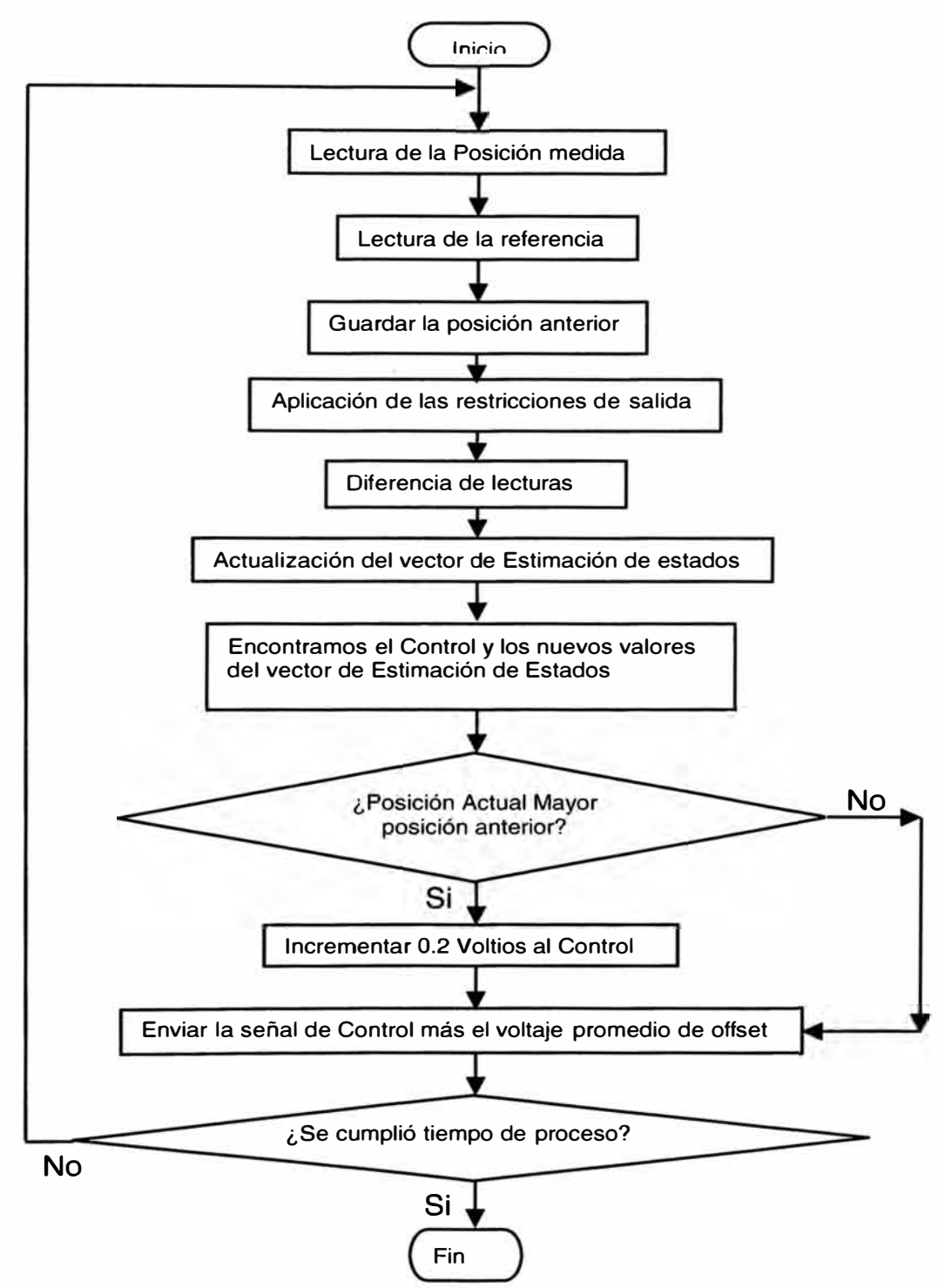

Figura 4.4: Diagrama de flujo del programa a implementar.

Diagrama de flujo de mejoramiento de la señal de control

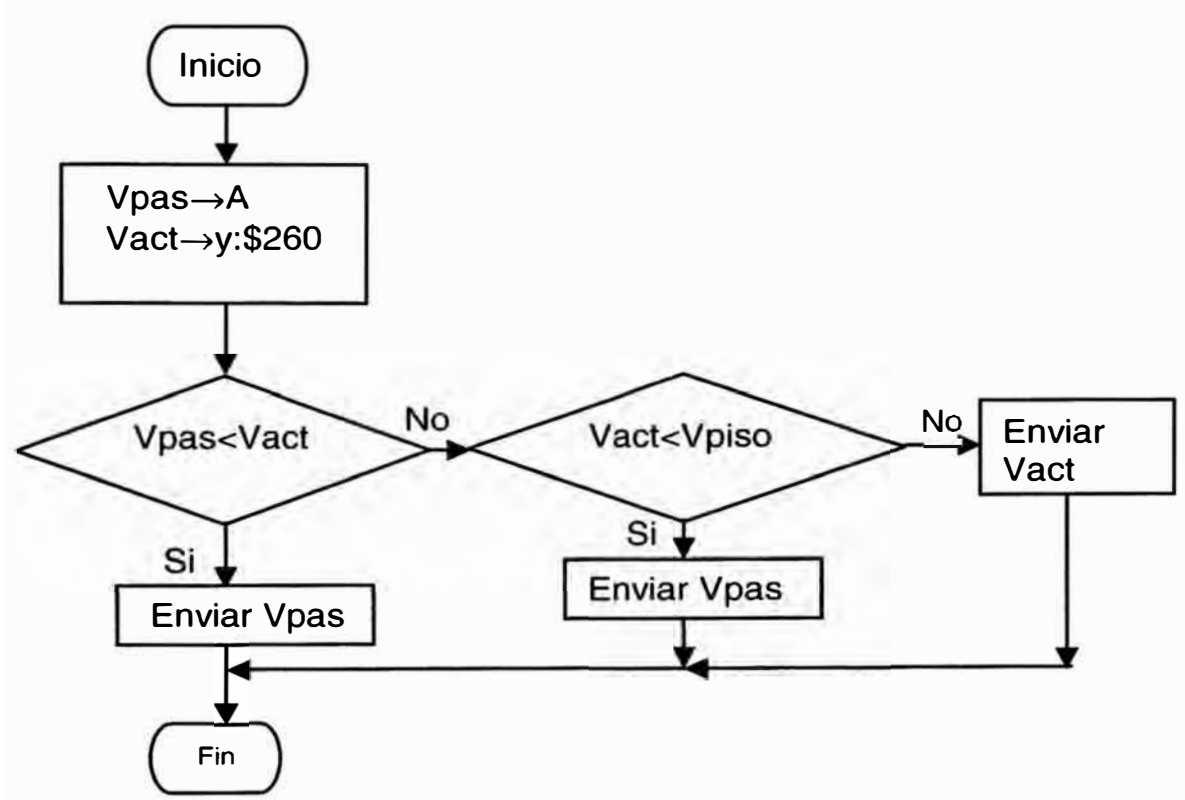

Figura 4.5: Diagrama de flujo de mejoramiento de la señal de control.

Diagrama de flujo del cálculo de la posición angular de salida

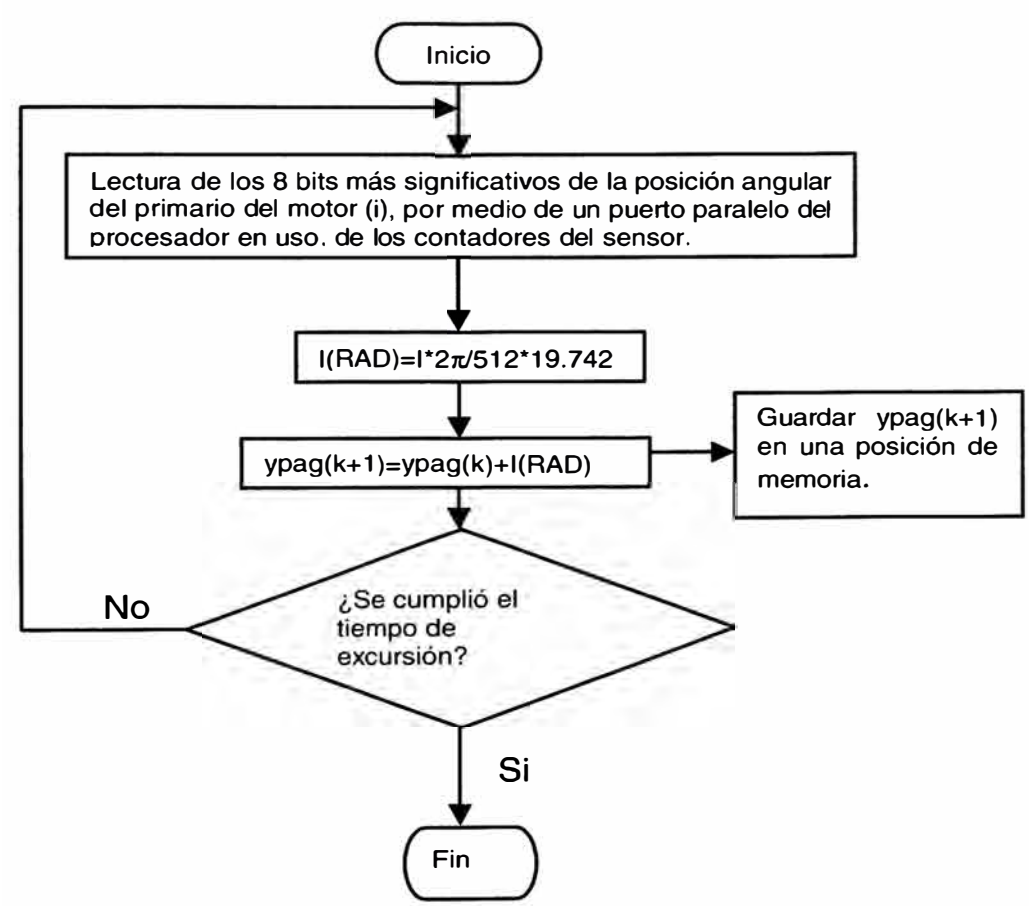

Figura 4.6: Diagrama de flujo para obtener la posición angular de salida.

CAPÍTULO V
HARDWARE DEL SISTEMA

En este capítulo se explican las partes físicas que componen el siste - ma de control de posición, funcionamiento y se destacan las simplificaciones importantes del método de mínimos cuadrados que disminuyen la cantidad de variables que es lo que se necesita para depurar los sistemas, sobre todo cuando de trata de implementar con un nuevo dispositivo, en este caso el DSP, que está en la fase inicial de sus implementaciones. El sistema de control de posición y la implementación física utilizadas están representados en la figura 2.1, en la cual se aprecian las siguientes partes:

- La planta del sistema de control de posición CAA.
- El sensor de posición.
- El generador de señal PWM.
- La tarjeta de adquisición de datos Lab-PC+.
- Una PC compatible con microprocesador Pentium.

Básicamente el funcionamiento es el siguiente: el detector óptico incremental genera dos series de pulsos en cuadratura cuya frecuencia es función de la velocidad del motor y su desfasaje función del sentido de giro. Estos pulsos son acondicionados, para alimentar contadores Up / Down estándares. La cuenta de 16 bits generada es proporcional a la posición del motor y es leída, a la frecuencia de muestreo, mediante dos puertos digitales de la tarjeta de adquisición. Calculado el valor de la posición, el software de control calcula la señal de control adecuada y la envía al generador de PWM mediante una de las salidas analógicas de la tarjeta. El generador de PWM (TTL) genera una señal de frecuencia constante y ciclo de trabajo proporcio - nal a la señal de control. La señal PWM alimenta el driver del motor, el cual genera una señal PWM de potencia la cual alimenta directamente al motor. Esta operación es realizada en cada periodo de muestreo.

5.1 La planta del sistema de control de posición CAA

La planta del sistema de control de posición auto-ajustable se presenta incluido en la figura 5.1 y está compuesto por:

Un motor de corriente continua PITTMAN de escobillas con reduc - ción de velocidad interna y un codificador óptico incorporado (que se explica- rá más adelante), la alimentación de este motor se realiza mediante dos ca - bles que salen del generador de señal PWM, la inversión en la posición de los cables no hace sino invertir el sentido de giro del motor.

Una varilla metálica de 77 centimetros de largo y de 64 gramos, que hace la función de carga no lineal.

5.2 El sensor de posición

El sistema posee un sensor que se encarga de medir la posición an - gular del primario del motor. Dicho sensor consiste de un codificador óptico rotatorio compuesto de un disco metálico con un número determinado de ranuras ubicadas en el perímetro de éste y un emisor-sensor óptico que genera una fuente de luz y sensa luego la presencia de ésta de acuerdo a su interrupción ó no-interrupción, el codificador incorporado en el motor es solidario a. ,e primario; es decir, para obtener el ángulo de giro en el eje de salida se le tiene que aplicar un factor de reducción cuyo valor está dado en la sección 1.3. El codificador óptico da como salida un tren de pulsos con una frecuencia proporcional a la velocidad angular del disco y un tren de pulsos desfasado +90 ó –90 grados respecto al primer tren, según sea el sentido de giro del disco, con mínimos cuadrados sólo se tiene un sentido de giro, con lo que únicamente necesitamos un tipo de desfasaje, para invertir el sentido de giro se puede utilizar números negativos en el intervalo [-1,0] ó invertir la posición de los cables de alimentación del motor. Estos trenes de pulsos pasan luego a un decodificador de cuadratura LS7083, el cual genera señales clock Up ó clock Down según sea positivo ó negativo el sentido de giro del disco, la simplificación será, tener un clock y para desfasaje Inter - cambiar los cables de alimentación como sea requerido por el sentido de giro. Las señales clock Up y clock Down alimentan la entradas Up / Down de un contador de 16 bits compuesto por cuatro contadores 74LS193 de 4 bits conectados en cascada, con un sentido de giro podríamos utilizar 8 bits los más significativos, que serán leídos en algún puerto paralelo del DSP.

Tenemos en conclusión que la cuenta almacenada en los contadores será una función lineal proporcional a la posición angular del eje del disco. Para obtener la posición angular a partir del número de pulsos dados por los con - tadores podemos usar la siguiente relación antes de la reducción:

- Para la posición angular del primario del motor = $2\pi/512$xnúmero de pulsos.

5.3 El generador de señal PWM

Alimenta al motor. Este amplificador está compuesto por un modula - dor PWM de baja potencia LM3524 que utiliza un fuente de 5 voltios TTL, una lógica digital de disparo y un conmutador amplificador tipo H implemen - tado con cuatro mosfets y una circuitería adicional que genera la tensión de disparo en cada conmutador mosfet, su función es la de amplificación de potencia, el procesador DSP envía por su puerto paralelo una señal modulada con una frecuencia portadora de 40 Megahertz al modulador PWM, como el DSP utiliza también 5 voltios la función de modulador PWM podría implementarse con el DSP, con lo que el sistema necesita solamente el conmutador amplificador tipo H. Con el método de mínimos cuadrados, se tiene un sentido de giro, son necesarios sólo 2 mosfets, en el amplificador tipo H, en conclusión se tendría un amplificador unidireccional con 2 mosfets.

5.4 La tarjeta de adquisición de datos

La tarjeta de adquisición de datos Lab-PC+ se utiliza para el envío de la señal de control y de adquisición de datos. La interconexión de la tarjeta de adquisición de datos con el amplificador se realiza de la siguiente forma:

- Puerto A: recibe los 8 bits menos significativos del sensor.
- Puerto B: recibe los 8 bits más significativos del sensor.
- Puerto C:

  -Bit 0: entrada de timer.
- DAC0: envía la señal de control.
- OUTB0: envía clock de muestreo.

Con el empleo del DSP, no necesitamos emplearla, aunque también es recomendable cuando la magnitud del programa en el DSP es extensa ó laboriosa, podemos usar los datos de salida de programa en C (en el cual cumplió su labor esta tarjeta) en la labor de simulación, para verificar la producción de la señal de control y corregir errores, para etapas de comparaciones de errores real y teórico, en donde si aceptamos un valor mínimo de diferencia de error y la relativa rapidez del proceso de discretización, podemos comprobar físicamente, colocando directamente a la salida del generador PWM (que nos está amplificando la señal producida por el DSP) al motor, en esta comprobación realizada, como un paso a la implementación en tiempo real, en que se pueda leer los bits del contador del sensor con un puerto paralelo del DSP ó usar la tarjeta Lab-PC+.

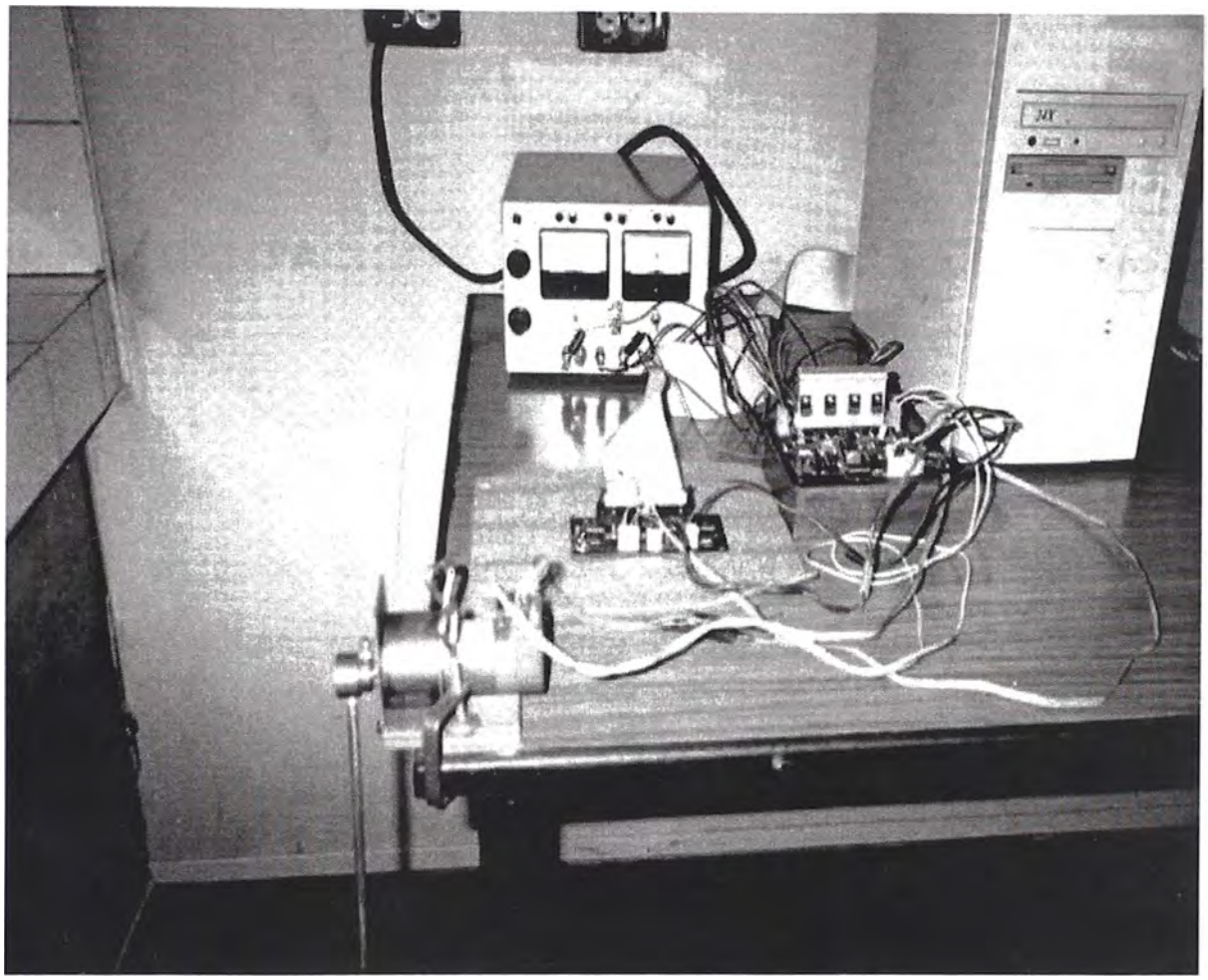

Figura 5.1: Sistema de control de posición auto-autoajustable (SCAA).

CAPÍTULO VI
IMPLEMENTACIÓN DEL SOFTWARE Y RESULTADOS

De alguna forma se contempló los diferentes requerimientos de los al - goritmos que necesita el control CCA, que se implementaron buscando siempre un grado satisfactorio de estabilidad y tratando de facilitar la elaboración del programa ensamblador que se encuentra en una fase inicial de implementación en el DSP. En primera instancia se confeccionaron programas de extensión "cpp" que son programas en C, en los cuales, sus salidas presentaban mucha distorsión, sobre todo el sobre impulso inicial era notorio, la vibración era otro problema que había que solucionar, restringien - do valores de entrada y salida se confeccionaron programas que iban suavi zando las señales hasta poder quitarles las condiciones de saturación de la señal de control u limitadas y volverlas a su condición original de 1.4V, el trabajo se centró en poder obtener el programa en lenguaje ensamblador, pero tan sólo en las gráficas de la salida de posición disminuíamos la altura del sobre impulso, que todavía seguía presente, pero las etapas del programa, que se empezaron a agregar porque fueron posibles, y que podrían ser implementadas en lenguaje ensamblador, fueron realizables en tiempo real, se inició obviando la etapa de IRLS, por lo extensa, dentro del dominio de números fraccionarios con que se trabaja en el DSP usualmente y en la que se cuenta con instrucción de multiplicación y acumulación MAC,

luego utilizando constantes de ganancia llamada de Kalman y calculada con software de Matlab para la estimación de estados sin etapa de IRLS, seguidamente conservando la ganancia utilizada para la estimación de estados también se calcula la ganancia para la etapa de control mediante el mismo software de Matlab, con los resultados anteriores y lo que se refería a restringir la salida como predicción era lo que de allí en adelante se desarrolla en el uso de la regresión lineal como en la referencia [13] que se relaciona con mínimos cuadrados y que tiene que ver directamente con la discretización, un mayor alcance de mínimos cuadrados se reserva para el Anexo A, se tiene una recta de regresión que en este caso la hacemos coincidir con la referencia como una muestra adecuada para llegar a la posición deseada sin sobrepasarla con los valores de salida de posición y ayudados por el hecho de que al colocar inicialmente 0.35 voltios menor de 0.7854 voltios la posición deseada, un valor menor de la posición deseada que no ocasione sobre impulso, las respuestas de la salida no sobrepasan a la referencia, luego de varios instantes posteriores al inicio del proceso, lo que permite que cuando la salida supere a la recta de la referencia antes de la mitad del primer segundo la salida se monta en la referencia que no ha llegado aún a la posición deseada pues la referencia está programada para llegar a la posición deseada al final del primer segundo evitando de esta forma el problema del sobre impulso, más ya que por un problema de números fraccionarios por la extensa cantidad de instrucciones de programa en lenguaje ensamblador que se utilizaría para controlar el sobre impulso, mínimos cuadrados es una forma de predicción que nos lleva a prevenir ó

precaver y adelantarnos a los procesos, esto condujo a un programa de control de posición de 20 segundos de muy buena respuesta, también se pudo controlar un seguimiento de 40 segundos de dos posiciones y un control de seguimiento de dos posiciones para 100 segundos; se estudia el control de dos posiciones porque inicialmente la recta va tomando diferentes posiciones antes de alcanzar la posición deseada, se trabajó con programas duales a los anteriores con etapa RLS pero con problemas de robustez para él de 100 segundos.

La robustez a la excursión de seguimiento de 100 segundos se obtuvo cuando se restringió el valor de la señal de control en valor absoluto, con buena respuesta aunque la brecha de tiempo tan grande no permitió que tenga efecto la regresión en el primer segundo de cada tramo, de aquí en adelante con estos resultados se completo el programa de lenguaje ensam - blador del DSP para el control de posición solamente, pues la labor de comunicación de la planta es extensa dicho programa de extensión asm como para todos los programas en lenguaje ensamblador del DSP. Para la implementación del algoritmo de control se procedió como sigue: primero se calculó la ganancia de estimación de estados (Kb) y la ganancia de la etapa de control (K) utilizando Matlab, lo que es optimizado, luego se les colocó en el programa para ser usadas por él para controlar al sistema. Para la lectura de los datos de salida, primero el programa los guarda en un archivo para luego ser leídos y analizados posteriormente empleando Matlab.

6.1 Elección de la frecuencia de muestreo

Se debe elegir una frecuencia tal que todas las instrucciones y cálculos del programa se realicen completamente, se elige un periodo de muestreo de Ts=0.01 segundos, en este tiempo la señal permanece cons - tante, como las señales eran aproximadamente semejantes en periodos más largos, se procedió por discretización a considerar un tiempo de 0.05 segundos, para actualizar las señales, estos tiempos son posibles de calcular, teniendo en cuenta el número de instrucciones del programa, la frecuencia de operación, y el teorema del muestreo, así para 27 Megahertz de frecuencia de operación, 5000 instrucciones de programa, el periodo mínimo de muestreo es de 400 microsegundos, con el DSP es posible con - seguir estas perfomances, dado que su frecuencia de operación normal de 4 Megahertz, se convierte a 40 Megahertz, programando adecuadamente un multiplicador, en el registro de PLL, para mantener un periodo de 0.05 segundos se coloca un número adecuado en el registro del contador del timer del DSP, este número se obtuvo con un osciloscopio tektronik de laboratorio.

6.2 Simulación del programa de control de posición usando Simulink

Para proceder a la simulación del algoritmo de control se usaron los siguientes datos que se dan en la tabla 6.1 (la obtención de las ganancias de estimación de estados y de control se hará posteriormente) y están en un archivo de extensión “m” de Matlab. El modelo del control de posición que utiliza los datos anteriores se muestra en la figura 6.1, para una entrada de escalón unitario como referencia, que se inicia en el tiempo cero, la

respuesta para un segundo de excursión se da en la figura 6.2; la planta en la figura 6.1 se ubica en el bloque " planta caa", el sistema de controlador y planta trabajan en el dominio discreto con un tiempo de discretización de 0.05 segundos (como se explica en la sección 6.1) y están separadas por un bloque de zero-order, que es el bloque que indica dominio discreto en Simulink.

6.3 Archivo de interfaz

Para facilitar el uso de la tarjeta de adquisición de datos se utilizó el archivo R1LLIB, que utiliza las librerías NIDAQ de la referencia [1] y funciones de lectura y escritura de puertos. Este archivo consta de las siguientes funciones:

Configurar hardware

Inicializa la tarjeta LAB-PC+, configura los puertos y especifica la frecuencia del clock de muestreo.

Enviar voltaje

Envía al puerto de salida el voltaje DAC0 del LAB-PC+.

Leer sensor

Adquiere datos del número de posiciones registrada en los contado - res, en los puertos A y B.

Nivel de clock

Devuelve "1" ó un "0" alternativamente con la frecuencia indicada en Configurar Hardware.

Datos	Valor	Descripción
G	$\begin{bmatrix} 0 & 1 \\ -0.6246 & 1.6246 \end{bmatrix}$	Es la matriz que representa la planta.
H	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$	Es la matriz de control.
C	$\begin{bmatrix} 0.0410 & 0.0479 \end{bmatrix}$	Es la matriz de salida.
K	$\begin{bmatrix} -0.5215 & 1.34760 \end{bmatrix}$	Parte de la ganancia de control.
K2	-0.5128	Completa la ganancia de Control.
Kb	$\begin{bmatrix} 1.10 \\ 1.16 \end{bmatrix}$	La Ganancia de Estimación de estados
R	7.38Ω	Es la resistencia de armadura.
Kt	0.031071	Ver Tabla 1.2.
L	0.00464 Henrios	Es la inductancia de armadura.
E	0.031071	Ver Tabla 1.2.
n	19.7411	Ver Tabla 1.2.
Jeff	0.0000563	Ver Tabla 1.2
beff	0.0000705	Ver Tabla 1.2.
m	0.06377 Kg.	Masa de la varilla.
Lo	0.776 m	Longitud de la varilla.
g	9.8	Aceleración de la gravedad.
Kac	14.9	Ver Tabla 1.2.
Ts	0.05 seg.	Periodo de muestreo.
C1	0.0064	Peor caso para fricción de Coulomb.

Tabla 6.1: Descripción de datos usados para la simulación del programa de control de posición en Simulink.

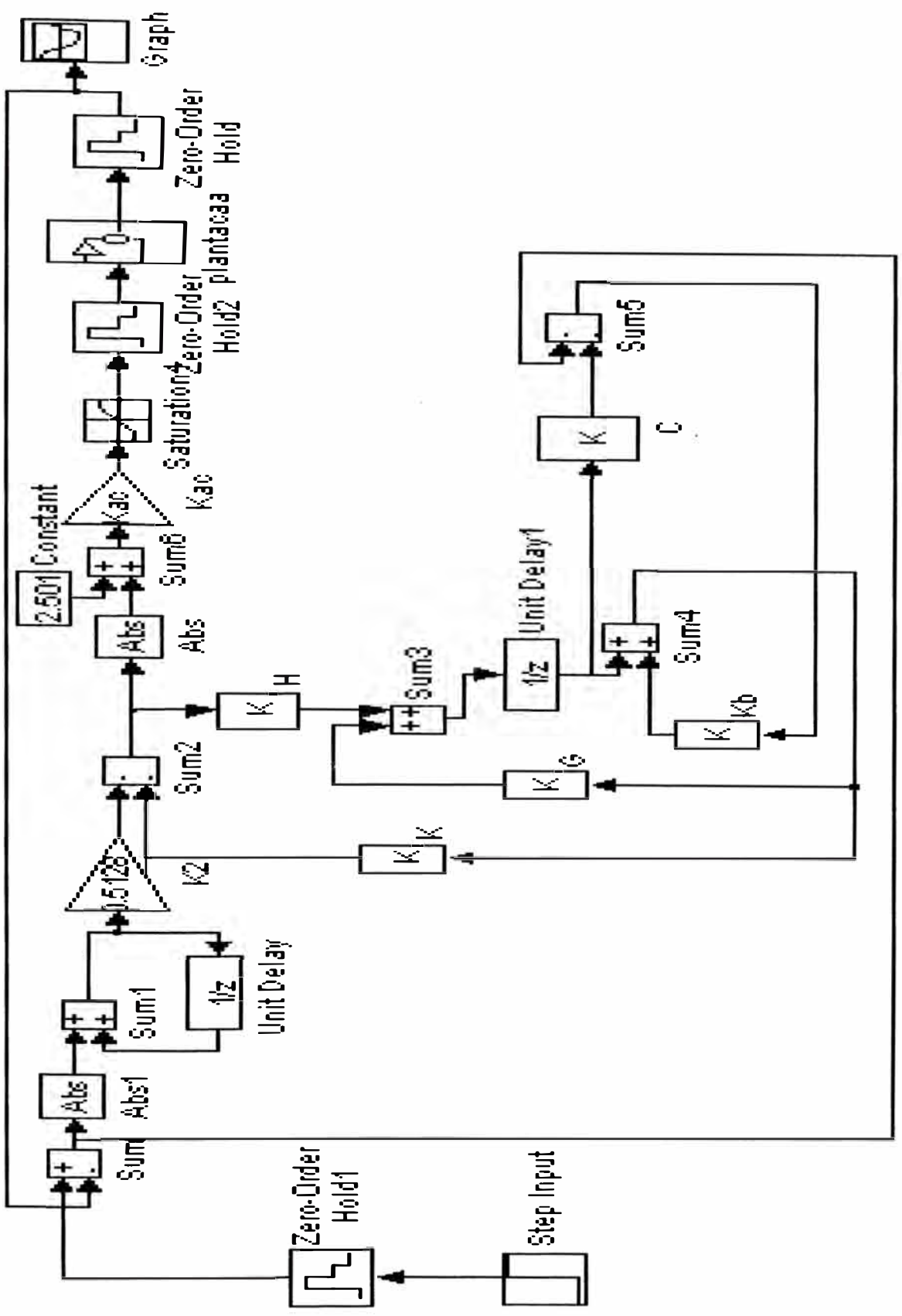

Figura 6.1: Diagrama del Modelo en Simulink del control de Posición.

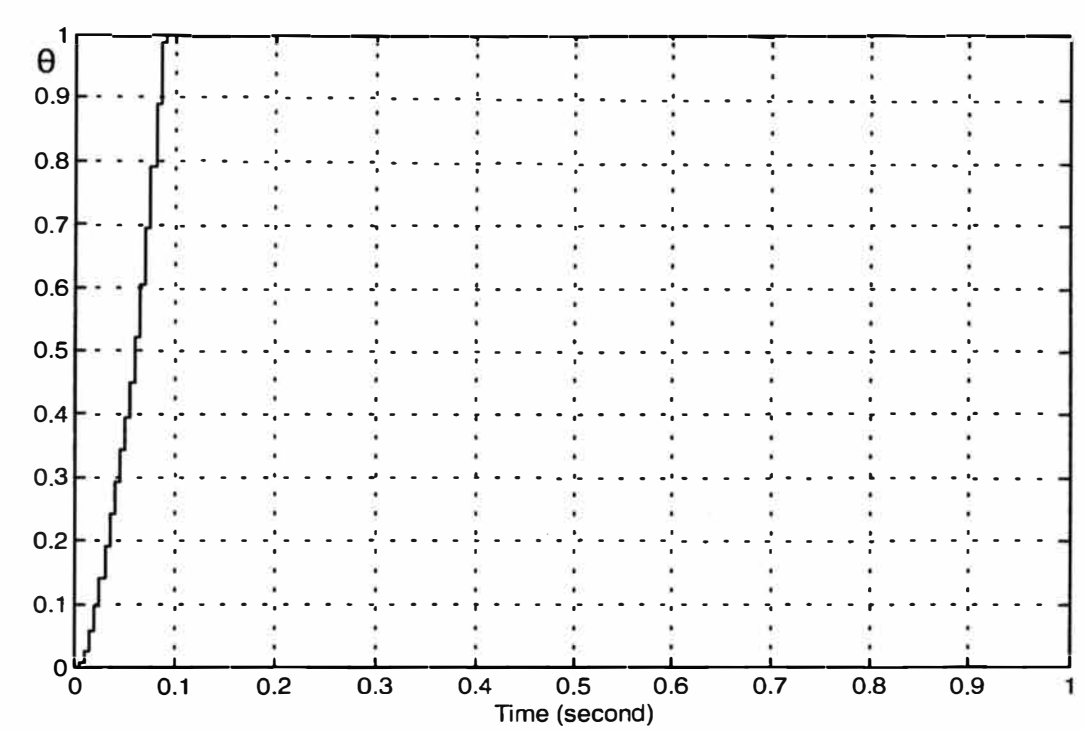

Figura 6.2: Respuesta del control de posición de la figura 6.1, a un escalón de referencia unitario.

6.4 Programa de control del control de posición CAA en C++

El programa consta fundamentalmente de las siguientes etapas:

1. Iniciación
   (a) Inicialización de la tarjeta de adquisición de datos.
   (b) Inicialización de las variables.
   (c) Establecimiento de la posición cero de la varilla.
2. Algoritmo de control
   (a) Detección del flanco de subida de clock.
   (b) Medición de las salidas.
   (c) Enunciado de la referencia.
   (d) Estimación de estados.
   (e) Cálculo y aplicación de la ley de control.

Iniciación de la tarjeta de adquisición de datos

Se configura la Lab-PC+ con la función configurarHardware(Fs), en donde se especifica la frecuencia del clock de muestreo Fs que se debe generar y las direcciones de memoria que se van a utilizar.

Inicialización de variables

Se inicializan todas las variables a utilizar en el programa y se cargan las matrices que previamente se calcularon utilizando Matlab. En el progra - ma la constante de estimación de estados, se calcula con software de Matlab, se utiliza el comando DLQE de Matlab para obtener la solución de la ecuación de Riccati de orden 2, la ganancia llamada de Kalman por ser obtenida con el filtro estacionario de Kalman discreto siendo de orden 2*1, permite actualizar las variables del observador. Para obtener la ganancia tenemos la siguiente ecuación:

$$[M,P,Z,E] = DLQE(G,H,C,Q,R) \qquad (6.1)$$

Donde: G, H y C son de la tabla 6.1, Q=0.01 y R=0.04 son halladas para mejor respuesta, con la aplicación de ecuación 6.1, en el ambiente de Matlab obteniendo la matriz de la constante de ganancia de estimación de estados Kb de orden 2*1 que es:

$$M = Kb = \begin{bmatrix} 1.10 \\ 1.16 \end{bmatrix}$$

Hallándose además la solución de la ecuación de Riccati que es una matriz de orden 2 que llamamos PB cuyos elementos son agregados al programa como constantes dicha matriz es:

$$P = PB = \begin{bmatrix} 0.5386 & 0.5605 \\ 0.5605 & 0.5986 \end{bmatrix}$$

En el Programa las constantes son calculadas con software de Matlab, se utiliza el comando DLQR de Matlab para obtener la solución de la ecuación de Riccati de orden 3, la ganancia llamada de control que se obtiene con realimentación que reduce al mínimo la función del costo sujeta a la ecuación de diferencias. Para obtener la ganancia de control tenemos la siguiente ecuación:

$$[K,S,E] = \text{DLQR}(GE,HE,Q,R) \qquad (6.2)$$

Donde GE, HE, Q y R son como se muestra a continuación:

$$GE = \begin{bmatrix} 0 & 1 & 0 \\ -0.6246 & 1.6246 & 0 \\ -0.041 & -0.0479 & 1 \end{bmatrix},\ HE = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix};\ R = 0.05$$

Obteniendo la matriz de la constante de ganancia de la etapa de control K de orden 1*3 que es:

$$K = \begin{bmatrix} -0.5215 & 1.3460 & -0.5128 \end{bmatrix}$$

Hallándose además la solución de la ecuación de Riccati, que es una matriz de orden 3 que llamamos P cuyos elementos se tuvieron que agregar al programa como constantes dicha matriz es:

$$S = P = \begin{bmatrix} 0.1185 & -0.0340 & -0.0540 \\ -0.0340 & 0.3303 & -0.1950 \\ -0.0540 & -0.1950 & 1.8088 \end{bmatrix}$$

Establecimiento de la posición cero de la varilla

Se coloca la varilla en la posición cero. La varilla debe colocarse en posición vertical y hacia abajo (péndulo convencional). Con esto el programa identifica la posición de 0º como inicial.

Detección del flanco de subida del clock

Detecta el flanco de subida del clock mediante la función NivelClock, lo cual nos especifica el comienzo de un nuevo periodo de muestreo.

Medición de las salidas

Para obtener la posición angular a partir del número de pulsos dados por los contadores, podemos usar la siguiente relación, antes de la reduc - ción:

Posición angular del eje del motor = $2\pi/512$xnúmero de pulsos.

Para la lectura de Los sensores tenemos que tomar en cuenta la saturación de los contadores, los cuales pueden guardar un valor máximo de 65536. Para este caso, Los contadores no llegan a saturarse, pues a lo sumo la varilla podría tener un recorrido de 360°, lo cual da un incremento de 512, que está bastante lejos de alcanzar el valor máximo del contador. El sistema puede reconocer si se está en un ángulo negativo si el valor del contador es mayor que 8000.

Enunciado de la referencia

En este programa la referencia es el mejor comportamiento para la posición de salida y esta se hace coincidir con la referencia, después de la estimación de estados, la cantidad en este caso la posición de salida nunca debe ser superada, esto cumpliendo la técnica de regresión lineal como en

la referencia [13] donde la cantidad es un polinomio que depende de variables independientes, el polinomio tiene como término independiente el valor anterior de la posición y las variables independientes dependen de la señal de control que aplicada al motor y varilla, representado estos por parámetros, el tiempo que sube la recta se obtuvo experimentalmente y fue de 1 segundo pues los de 0.5 segundos ó 1,5 segundos que desestabilizó a la planta, que no controlaba, el valor inicial de la referencia, mayor que cero se ajustó en 0.35. El programa implementado que es la culminación de otros que lo precedieron, se caracteriza por el inicio en el empleo de la referencia variable y el ajuste de la posición a la referencia en cada ciclo de discretiza - ción, para una duración de 20 segundos. La referencia variable utilizada en este programa se tiene en la siguiente ecuación:

$$r(t) = \begin{cases} 0.35 + 0.435398t & \text{si} \quad t < 1.0 \ \text{seg.} \\ 0.785398 & \text{si} \quad 20. \ \text{seg.} > t \geq 1.0 \ \text{seg.} \end{cases} \tag{6.3}$$

Estimación de estados

El planteamiento de las ecuaciones para la estimación de estados es:

$$rr = y - C[0] * x[0] - C[1] * x[1] \tag{6.4}$$

$$x[0] = x[0] + Kb[0] * rr \tag{6.5}$$

$$x[1] = x[1] + Kb[1] * rr \tag{6.6}$$

$$z = z - y \tag{6.7}$$

Donde: y, es la diferencia entre la lectura de la salida y el valor de la referencia. rr, es la diferencia de y con el valor estimado para y. $\hat{\vec{x}}$, es la

matriz de estados estimados de orden 2*1. Kb[0] y Kb[1], son los elementos de la ganancia de estimación de estados. Z, es la variable resultante de la acción integral.

Cálculo y aplicación de la ley de control

El planteamiento de las ecuaciones para la etapa de control es:

$$u = -(K[0] * x[0] + K[1] * x[1] * K[2] * z) \quad (6.8)$$

$$Raux1 = G[0][0] * x[0] + G[0][1] + x[1] + H[0] * u \quad (6.9)$$

$$x[1] = G[1][0] * x[0] + G[1][1] * x[1] + H[1] * u \quad (6.10)$$

$$x[0] = Raux1 \quad (6.11)$$

Para cumplir con los requerimientos del método de mínimos cuadra - dos, tomamos el valor absoluto de la señal de control como sigue:

```
If(u<0 )                u=(-1.)*(fabs(u));
else if(u>0.0)          u=fabs(u);
*U=u;
```

Para enviar aplicar la señal de control utilizamos la siguiente parte de programa:

```
/*Aplicación de control*/
float v, wth, ofst;   /*wth=.1; v=.2, ofst=2.3;
float aplica_control(float u, float ykp)
{ float v;      if(u+v>1.4)              u=1.4;
 Enviar voltaje(u+v+ofst);
  return u; }
```

La respuesta del sistema, se muestra en la figura 6.3, donde se tiene de arriba para abajo: la referencia, la señal de control, la posición y el error, aquí la estabilidad manifiesta con el valor absoluto tomado, las señales tienen el mismo sentido, se tuvo una mayor robustez del sistema, pudiéndo - se agregar cargas. La figura de la implementación física, donde se observa el resultado, se tiene en la figura 6.4.

6.5 Programa de simulación del control de posición CAA en ensam - blador del DSP

El programa consta fundamentalmente de las siguientes etapas:

1. Iniciación
   (a) Inclu}sión de los programas de datos.
   (b) Inicialización de las variables.
   (c) Establecimiento de la posición cero de la varilla.
2. Algoritmo de control
   (a) Subrutina para discretizar en el tiempo de muestreo.
   (b) Medición de las salidas.
   (c) Enunciado de la referencia.
   (d) Estimación de estados.
   (e) Cálculo y aplicación de la ley de control.

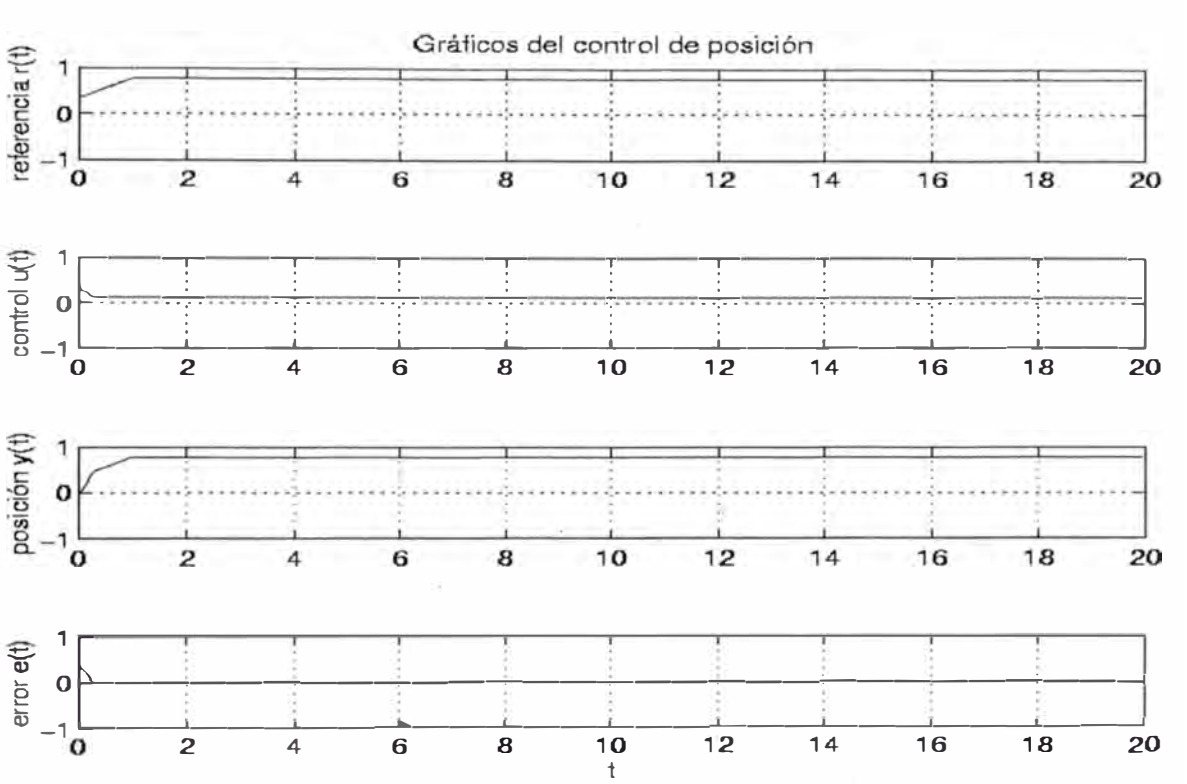

Figura 6.3: Gráficos del control de posición del programa implementado en C++.

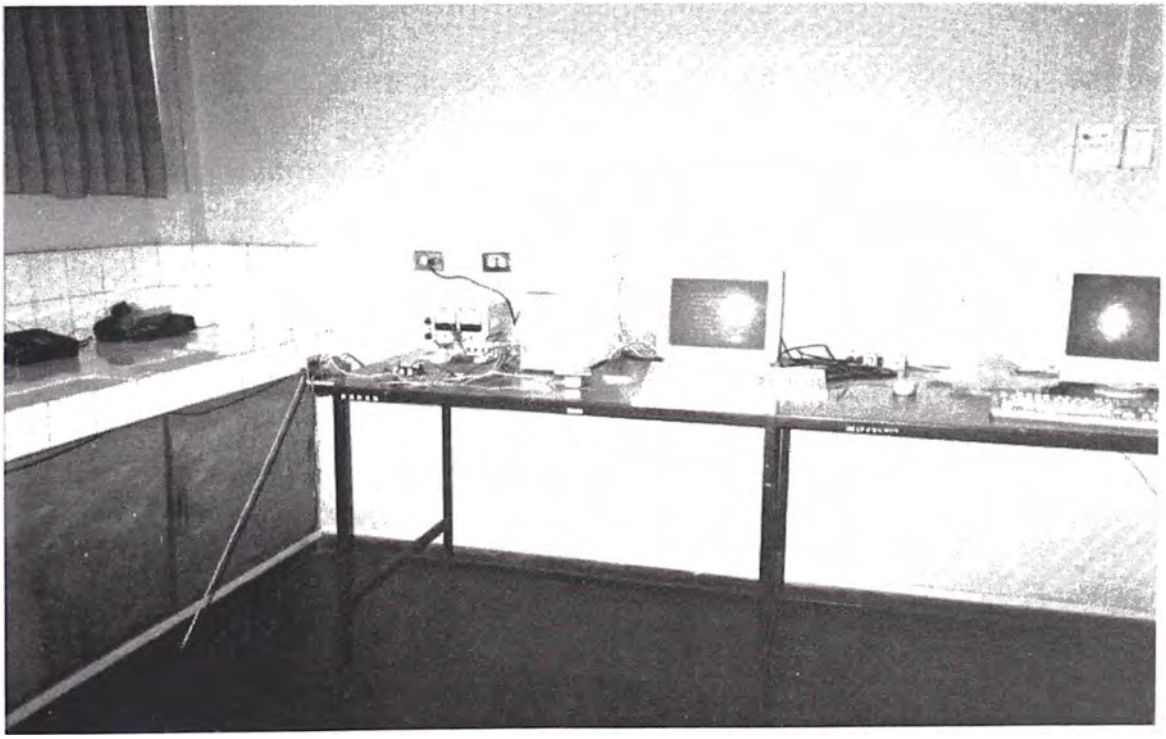

Figura 6.4: Implementación física del sistema mostrando el resultado de la posición deseada.

Inclusión de los programas de datos

Utilizamos los datos del archivo de salida del programa en C++, elaborado con este objeto, como modelo de referencia, se incluye como cabecera al inicio del programa, un archivo con la instrucción "include", que reconoce datos desde el inicio del programa, este archivo puede incluir los datos de la posición de salida, que es la tercera columna del archivo de salida del programa en C++, la referencia se puede incluir por programa, y el error que es la diferencia entre la referencia y la posición de salida, que constituye la quinta columna del archivo de salida del programa en C++, es este error que debemos controlar, que nunca tome valores negativos, para cumplir con la técnica de mínimos cuadrados, tomando el valor absoluto del error conseguimos este objetivo, si se trata de simplificar también es posible crear un archivo de cabecera con la quinta columna del archivo de salida, como se hizo en la simulación, la forma en que se incluyo dicho archivo de cabecera, lo mostramos a continuación:

include'ycomp1.asm'.

Inicialización de las variables

Se inicializan todas las variables a utilizar en el programa, se declaran todos los registros importantes que se van a utilizar, todas las variables que se emplean en posiciones de memoria, se procede a inicializar los registros declarados y se cargan las matrices que previamente se calcularon utilizan - do Matlab como se estableció en la sección 6.4.

Establecimiento de la posición cero de la varilla

Se procede al igual que la sección 6.4.

Subrutina para discretizar en el tiempo de muestreo

El programa utiliza una subrutina de espera a interrupciones, previa - mente en el contador del timer (tcr) se coloca un número que discretiza el programa a 0.05 segundos, trabajando con una frecuencia de operación de 40 Megahertz, y utilizando un osciloscopio de laboratorio, se encuentra que dicho número hexadecimal es f4240, la subrutina que hace esto se tiene a continuación:

```
in1     jsr             xm1
        jclr            #07,x:tcsr,*
        jmp             in1
xm1     movep           y:(r4)+, x:pbd
        jset            #08, x:pbd, pua
        bset            #$00, x:tcsr
        rts
pua     jmp *
```

Medición de las salidas

Cada periodo de discretización el programa actualiza el valor de la posición de salida, desde el registro de cabecera, luego que se actualice la referencia por programa, se procede al cálculo del error, que es el valor de interés.

Enunciado de la referencia

La referencia de la ecuación (6.3), de la sección (6.4), se obtiene en ensamblador del DSP por programa, primero se inicializa (0.35), y luego cada vez que se cumpla el tiempo de muestreo se le agrega un incremento constante, hasta que en el lapso de 1 segundo se ha llegado a la posición deseada (0.785398), a partir de este momento se mantiene a la referencia constante.

Estimación de estados

Para poder utilizar la instrucción mac, se dividieron por programa las cantidades entre 100, lo que no afecta los resultados, procediendo adecua - damente para recuperarlos, multiplicando por 100 ó utilizando corrimiento de bits. Primero se calculó rr, después x[0] y x[1], los elementos de la matriz de estados, tomando en cuenta sus valores en el instante de muestreo anterior, finalmente se calcula z y se tiene cuidado con la aproximación, para que los datos pasen a la etapa de control. La ganancia de estimación de estados, se uso directamente en el programa y es la misma que en la sección 6.4.

Cálculo y aplicación de la ley de control

Con los valores de x[0], x[1] y z obtenidos en la estimación de parámetros, que se colocan en posiciones de memoria, al calcularlas en cada periodo de discretización, se procede a efectuar el algoritmo de la etapa de control, dados en la ecuaciones (6.8), (6.9), (6.10) y (6.11), procediendo a realizar multiplicaciones, es aquí que utilizamos la instrucción MAC, se calculan los nuevos elementos de la matriz de estados, ajustados los cálculos y efectuados los corrimientos de bits, se posiciona el valor de la

señal de control para enviarlo por un puerto paralelo, considerando los 8 bits más significativos de la palabra de 24 bits, que se mantendrán en el puerto paralelo por lo que dure el periodo de discretización, los 8 bits que representan a la señal de control, se convierten a su valor análogo con el uso de un convertidor digital para analógico de 8 bits, ver figura 2.10, y de allí, adicionándole la señal de offset de 2.3 voltios enviadas al generador PWM.

La respuesta de simulación del sistema de control de posición mediante DSP, con e mejoramiento de señal como se manifiesta en sección 4.2.3, se muestra en la figura 6.5, la prueba física dio como resultado, igual que el presentado en la figura 6.4.

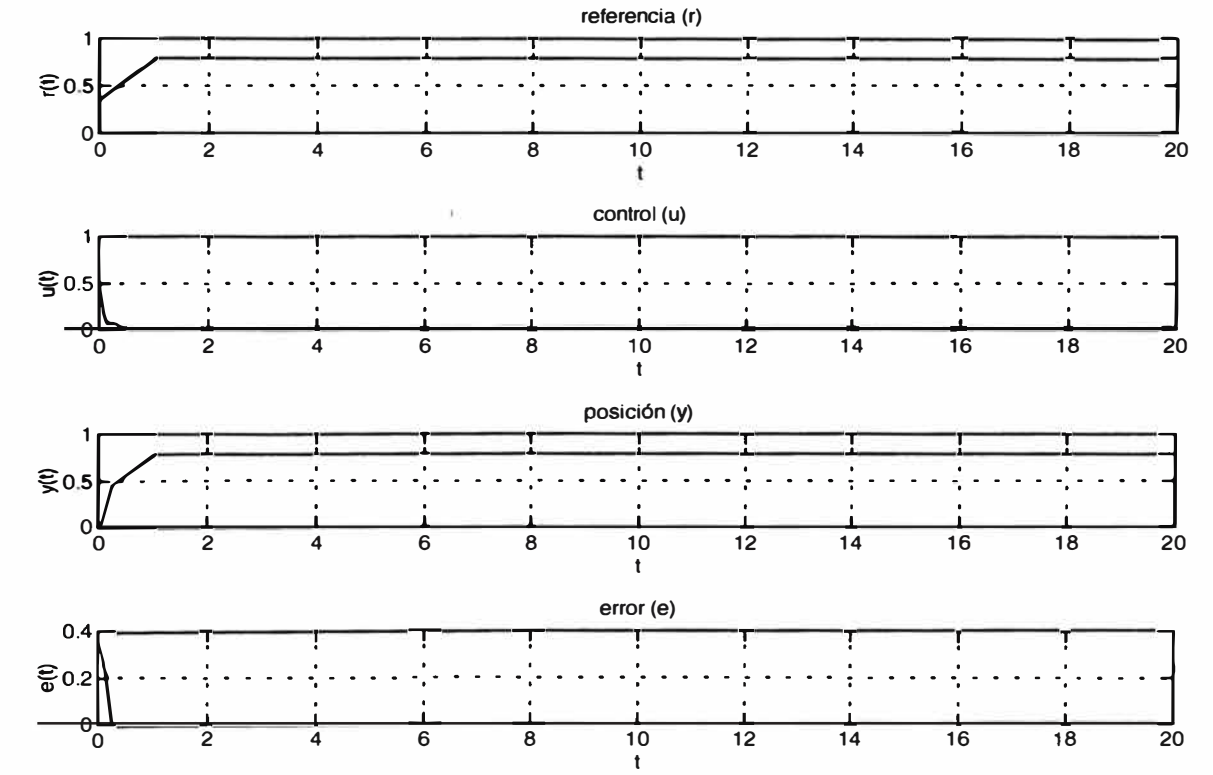

Figura 6.5: Respuesta de simulación del sistema de control de posición CAA mediante DSP.

6.6 Resultados de la simulación y sus implicancias

Los tres principales objetivos de esta tesis han sido: utilizar el DSP56002 para realizar un proyecto en el área de control, estudiar el aspecto de la implementación del SCAA con el DSP partiendo de un sistema ya implementado y tener un programa que permita evaluar las posibilidades de trabajar con algoritmos en el DSP.

Para lograr el primer objetivo, se tuvo en cuenta el criterio de optima - bilidad de los controladores para lo cual se implementó programas en C++ en los que de acuerdo al número de instrucciones se escoge el tiempo de muestreo tal que todas las instrucciones tengan efecto en dicho tiempo, con la factibilidad de dichos programas en C++, era posible iniciar el proyecto de implementación del programa del control de posición de un motor de co - rriente continua y carga no-lineal, adaptivo, autoajustable y discreto, se corrigió el problema de excesiva vibración impidiendo el sobre impulso del primer segundo con la aplicación de la regresión lineal, el sistema es práctico al no usar cabeceras que alargan los programas, además de no utilizar un lenguaje artificioso, bastándonos lo más substancial como cálculos de estimación de estados , de la señal de control y el uso de una subrutina de espera a interrupción para establecer el tiempo de discretización, lo que es independiente del número de instrucciones. Utilizando código hexadeci - mal obtenemos una importante simplificación como la de poder agregar la señal de offset en forma numérica como una implementación de instrucción de programa, aunque ello escapa del campo fraccionario que utilizamos, es posible expresar 1.15 como el número hexadecimal de 8 bits 94 consideran -

do que este número ocupa la posición de los 8 bits más significativos de los 48 bits del acumulador, aunque esto es la mitad del valor usado como offset, podemos utilizar un amplificador por dos y todavía no se logra saturar el amplificador PWM (logrando ahorrarnos el sumador análogo externo), y así también tener 1.88 como F1, sólo se puede expresar hasta 2.0 con nuestros 8 bits, luego al tener un número hexadecimal mayor tendremos a su vez mayor corriente de ponderación, si deseamos valores mayores de 2 pode - mos usar el peso de 9 bits. El programa estudiado cumpla con lo propuesto y su salida de datos fue utilizada como modelo de referencia teórica, en los sucesivos pasos del programa en lenguaje ensamblador del DSP. Los puntos de comunicación de la planta son los puntos de mayor cantidad de ruido para abordar el problema siempre se combate el peor caso pues eso le da mayor consistencia al sistema, lo que constituye el segundo objetivo de tener el sistema SCAA, los resultados de enfrentarlo permiten afirmar lo siguiente:

1. La comunicación ya sea con el generador PWM ó con la salida del sensor de posición es factible utilizando puertos paralelos como lo demues - tran los estudios de la tarjeta de adquisición de datos Lab PC+ de National Instruments.

2. Utilizando una implementación de voltaje offset externo es posible obtener la señal de control total mediante la utilización de un sumador análogo externo Ver figura 2.11.

3. Para enviar la señal de control se tiene que se puede usar cualquiera de los 2 puertos paralelos el B ó C, utilizando los 8 bits menos

significativos y corrimiento de bits pues utilizamos 24 bits para realizar los cálculos, lo mismo para recibir la lectura de la posición, considerando la lectura de los 8 bits significativos, mediante el corrimiento de estos bits a las posiciones más significativas de los 24 bits empleados para realizar los cálculos en nuestra abstracción de simulación de esta operación .

4. El método de mínimos cuadrados aplicado a la señal de error como la diferencia de la señal de referencia y la lectura de posición en cada instante de discretización permite converger a la posición deseada dentro del tiempo ponderado de 1 segundo (0.25 segundos en la aplicación) de las pruebas en tiempo real y lenguaje C, como aplicación de estimaciones de evaluación de sobre impulso en pruebas en Matlab, esto quiere decir minimizando el error hasta hacer coincidir la posición con la referencia dada, dentro de límites razonables de lectura de posición, se puede afirmar que la referencia de señal de error evaluada en tiempo real es aceptable como corrección por el método de mínimos cuadrados para evaluación de la simulación del sistema de control en el DSP.

5. Por lo tanto, es posible tener el sistema SCAA con el procesador digital de señales DSP, porque podemos satisfacer los requerimientos anteriores, además su capacidad para trabajar en aritmética de punto fijo como lo demuestra el programa implementado.

Para cumplir el tercer objetivo, se implementó el programa en lengua - je ensamblador, utilizando instrucciones MAC se simplificó los cálculos, con posiciones de memoria especificas se ahorra instrucciones de programa, con la aplicación de mínimos cuadrados a la señal de error, ósea minimizando el

error en valor absoluto, además con la señal de control en valor absoluto y su posterior refinación, el programa fue que pudo probar de ser factible a realizar pruebas que podrían indicar la inmunidad al ruido del DSP y a la que están sometidas las llamadas evaluaciones en tiempo real en lenguajes de alto nivel como C++ que fue la utilizada. La deducción de la naturaleza de las señales y la robustez dada por la convergencia del error debido al ruido, fue una larga investigación, las pruebas en DSP son más directas y simplifi - cadas, pudiendo resolver más rápidamente las interrogantes para tener un control de posición con mayor libertad en la elección de la posición deseada y la obtención de un programa general utilizando los datos obtenidos en la investigación.

CONCLUSIONES Y RECOMENDACIONES

Existe un amplio y vasto camino de investigación en este campo, profundizar controles de posición y seguimiento basados en mejor trata - miento y comprobación de la señal de error y señal de control, realización de pruebas más duras de comunicación en el DSP, ampliar las excursiones a ángulos negativos comprobar que debemos tener señales de error y control con el mismo signo. Unos ejemplos de extensiones de este trabajo se men - cionan a continuación:

1.- Este estudio sólo utilizó una señal de control de un convertidor digital para analógico más un offset externo, utilizar componentes más internos del DSP.

2.- Para la simulación se consideró los datos de salida del archivo de salida del programa en C++, graficándolos utilizando Matlab, para excursión de 45° utilizar otras salidas de datos y verificar hasta que punto puede existir proporcionalidad para los datos de salida, si con datos de posición de 45° podemos tener las posiciones de salida para 90° y 22.5° por ejemplo, esto se comprobó únicamente como cálculo numérico y podría conducir a un pro grama general.

3.- La utilización de otros dispositivos CMOS que permitan variar los valores de fuente de alimentación sin tener que someterse a los 5 voltios estrictos de la familia TTL.

4.- La profundización de las aplicaciones del temporizador (timer) del DSP para ahorrar etapas de programa como medida de periodos y generar señal PWM.

5.- Para comunicación con la microcomputadora debemos probar todos los medios que comunicación que dispone el DSP para aplicarlos a los sistemas de control.

6.- Implementación de controles de diferentes escalones de posición subiendo ó bajando por ejemplo entre 20° a 90° como un brazo robótico.

7.- Implementación de programa para corrección y ajuste de la posición deseada usando preferentemente el puerto B.

Los controles de posición y seguimiento utilizando el sistema de control CAA con tarjeta de procesamiento digital son una cosa cierta y práctica que no hay que menospreciar, los logros que se podrían obtener con su investigación son para tomarlos en cuenta, como alternativa de un futuro laboratorio de control.

ANEXO A
MÍNIMOS CUADRADOS

ANEXO A
MÍNIMOS CUADRADOS

A.1 Formulación de la ecuación

Considerando el modelo siguiente referenciado en la figura A1.

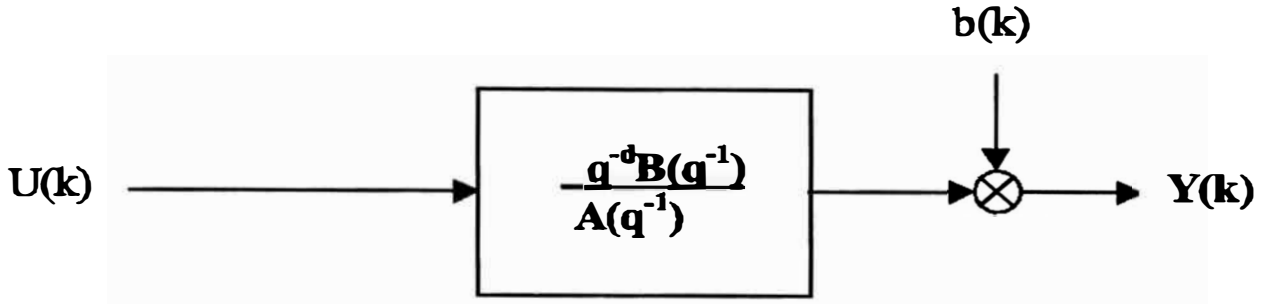

Figura A1: Descripción del modelo parámetrico en términos de entrada y salida del proceso.

Donde:

$A(q^{-1})=1+a_1q^{-1}+.....+a_nq^{-n}$

$B(q^{-1})=b_0+b_1q^{-1}+.....+b_nq^{-n}$

q^{-1} es el operador puro de atraso.

$(y(k).\ q^{-1}=y(k-1))$

d es el tiempo de retraso.

b(k) es el ruido blanco, incluyendo la medida de ruido del sistema.

El orden y retraso del sistema se suponen conocidos. La ecuación de diferencias puede ser obtenida de la ecuación siguiente:

$$A(q^{-1}).y(k) = B(q^{-1}).u(k) + A(q^{-1}).b(k) \qquad (A.1)$$

De donde:

$$y(k) = -a_1.y(k-1) - \ldots - a_n.y(k-n) + b_0.u(k-d) + \ldots + b_n.u(k-d-n) + e(k) \qquad (A.2)$$

Donde: e(k) es el ruido generalizado ó ruido residual y está expresado como:

$$e(k)=A(q^{-1}).b(k)$$

La ecuación (A.2) puede ser expresada utilizando notación matricial en la forma:

$$y(k) = \quad (k).\ (k) + e(k) \qquad (A.3)$$

Donde:

$\Psi(k)=[-y(k-1)\ldots \quad -y(k-n) \quad u(k-d)\ldots \quad u(k-d-n)]$

$\theta^T(k)=[a_1\ldots a_n \quad b_{0\ldots}b_n]$

$\theta^T(k)$ es la transpuesta del vector $\theta(k)$.

Usando N medidas de entradas y salidas la ecuación (A.3) puede generalizarse de la siguiente manera:

$$y = \phi.\theta + E \qquad (A.4)$$

Donde:

$$y^T = [y(n+1)\ldots \quad y(n)]$$

$$q^T = [a_1\ldots a_n b_0\ldots b_n]$$

$$E^T = [e(n+1)\ldots e(N)]$$

$$\varphi = \begin{bmatrix} -y(n)\ldots -y(1) & u(n-d+1)\ldots u(1-d) \\ \ldots\ldots & \ldots\ldots \\ -y(N-1)\ldots -y(N-n) & u(N-d)\ldots u(N-d-n) \end{bmatrix}$$

A.2 Estimación de los parámetros del modelo

Si los valores anteriores de entrada y salida del sistema son conoci - dos y si asumimos que el vector del modelo de parámetros θ es el mismo que el vector de estados actual del sistema, la mejor predicción del modelo de estado de salida esta dado por:

$$\hat{y} = \Psi(k).\hat{\theta}(k) \qquad (A.5)$$

Claramente, esto representa el caso ideal pero la situación real es diferente: primero, porque las medidas son contaminadas por el ruido y en segundo lugar, porque el modelo de vector de parámetros difiere del vector de parámetros actual del sistema. Esto ser puede expresar como el error de la predicción ó la ecuación del error:

$$e(k) = y(k) - \hat{y}(k) \qquad (A.6)$$

La calidad de la aproximación de un sistema real por uno de modelo paramétrico, se puede expresar como la variación de la predicción del error alrededor de su valor medio. Procuremos reducir al mínimo esta variación, eligiendo el criterio definido por:

$$C = \sum_{k=n+1}^{N} e^2(k) = E^T.E \qquad (A.7)$$

Esta ecuación puede ser expresada también de la forma:

$$C = (Y - \phi.\theta)^T.(Y - \phi.\theta) \qquad (A.8)$$

Este criterio puede ser minimizado como una función de la parametrización del vector buscando una manera tal que:

$$\frac{\partial C}{\partial \hat{\theta}} = \left[\hat{\theta} - \left[\phi^T.\phi\right]^{-1}.\phi^T.Y\right]\phi^T.\phi\left[\hat{\theta} - \left[\phi^T.\phi\right]^{-1}.\phi^T.Y\right] = 0 \qquad (A.9)$$

La solución a la expresión anterior esta dada por la menor estimación de los cuadrados. Así:

$$\hat{\theta} = \left[\phi^T.\phi\right]^{-1}.\phi^T.Y \qquad \text{(A.10)}$$

Esta solución se utiliza en situaciones, donde la opción del proceso fuera de línea es viable. Los vectores ϕ e Y contienen todos los pares de medida registrados del sistema al instante presente. Uno puede agregar todas las medidas disponibles, más exacta sería la estimación de vector de paráme - tros θ. Una solución alternativa es usar ecuaciones que computen secuen - cias de estimaciones del vector de parámetros $\theta(k)$ cada vez que son medidos. Este método conduce a las ecuaciones recurrentes que son fácilmente utilizables en aplicaciones de línea. Un conjunto típico de tales ecuaciones son:

$$\hat{\theta} = \hat{\theta}(k-1) + F(k-1).\psi(k-1).\Gamma(k)$$

$$F(k) = \frac{1}{\lambda_1(k)}\left[F(k-1) - \frac{F(k-1).\psi(k-1).\psi^T(k-1).F(k-1)}{\frac{\lambda_1(k)}{\lambda_2(k)} + \psi^T(k-1).F(k-1).\psi(k-1)}\right]$$

$$\Gamma(k) = \frac{y(k) - \hat{\theta}^T(k-1)\,\psi(k-1)}{1 + \psi^T(k-1).F(k-1).\psi(k-1)} \qquad \text{(A.11)}$$

Donde:

- $0 < \lambda_1(k) \leq 1$
- $0 \leq \lambda_2(k) \leq 2$
- $F(0) > 0$

- $\Psi(k-1)$, es el vector de medidas de datos (es decir el vector que contiene los datos de las medidas adquiridas del sistema) que es actualizado en cada ciclo del proceso.
- $\Gamma(k)$, representa el error de ajuste anterior del sistema.
- $F(k-1)$, representa la ganancia de ajuste. La estructura de este término es general.

Dependiendo la opción de secuencias $I1(k)$ y $I2(k)$, es posible tener un algoritmo autoajustable con:

- Ganancia de decrecimiento para $I1(k)=I2(k)=1$ (esta opción se utiliza para identificar sistemas inmóviles).
- Factor de olvido, para:

  $I1(k)=I(k)$

  $I2(k)=1$

- Constante de seguimiento, para:

$I1(k)=CI2(k)$ donde $C > 0$ (las opciones anteriores para $\lambda_1(k)$ y $\lambda_2(k)$ se utilizan para identificar sistemas que no varían en el tiempo).

- Ganancia proporcional, para:

  $\lambda_1(k)=1$

  $\lambda_2(k)=0$

A.3 El estimador de mínimos cuadrados

En orden a simplificar la notación en lo siguiente utilizaremos el estimador de mínimos cuadrados denotándolo como LSE dado por ecuación (A.10) y conveniente en aplicaciones fuera de línea.

A.3.1 Confianza en el método LSE

Recordemos que un estimador es estadísticamente posible sí:

$$E_M(\hat{\theta}) = 0 \qquad \text{(A.12)}$$

Donde: $E_M(x)$, es la esperanza matemática de x. Este valor medio de la estimación del vector de parámetros, converge al valor verdadero del vector de estados del sistema. Para el estimador LSE tenemos:

$$E_M[\hat{\theta}] = E_M[[\phi^T.\phi]^{-1}.\phi^T.Y] = \theta + E_M[[\phi^T.\phi]^{-1}.\phi^T.E] \qquad \text{(A.13)}$$

De aquí concluimos que el estimador LSE en cambios de polaridad, introduce un error en la estimación del vector θ.

A.3.2 Exactitud del estimador LSE

La aproximación de los cálculos hechos por el estimador es determinada por la variación del vector θ alrededor de su vector medio, es decir por evaluación de la matriz de varianza y covarianza simbolizada por $C_{\theta\theta}$. La exactitud del estimador LSE es dada por los elementos de la matriz denotados $C_{\theta i\theta i}$. La matriz viene dada por:

$$C_{\theta\theta} = E_M[[\hat{\theta} - \theta][\hat{\theta} - \theta]^T] =$$
$$= E_M[\phi^T.\phi]^{-1}.\phi^T.E.E^T.\phi.[\phi^T.\phi]^{-1}] \qquad \text{(A.14)}$$

Con respecto a la variación de los coeficientes de la planta y a la exactitud del estimador. Un estimador ideal se caracteriza por la variación nula de los coeficientes de la planta y mínima varianza. Un análisis de ecuación (A.14) y ecuación (A.13) indican que el estimador LSE manifiesta un cambio aunque pequeño de los coeficientes de la planta y que la matriz $C_{\theta\theta}$ no es mínima.

A.4 Comprobación del estimador LSE

Se obtiene considerando e(k) generalizado cumple con las siguientes propiedades:

$$E_M[\phi, E] = 0$$
$$E_M[E] = 0 \qquad \text{(A.15)}$$

De las ecuaciones (A.15) se entiende como que no debe haber correlación entre e(k) y Ψ(K) y que e(k) debe tener una distribución simétrica. La aplica - ción de estas características ecuación (A.13); nos retorna a la ecuación (A.12). El valor medio del vector estimado es menor ó igual en estas condi - ciones al vector de parámetros buscado.

A.4.1 Requerimientos para tener el mínimo estimador LSE

La variación del LSE se reduce al mínimo si se cumple con la siguien - te propiedad:

$$E_M[E^T.E] = \alpha^2 \qquad \text{(Varianza)}$$
$$E_M[E^T.E] = 0 \qquad \text{(Covarianza)} \qquad \text{(A.16)}$$

De estas ecuaciones se entiende que e(k) representa el ruido blanco. Apli - cando esta característica a la ecuación (A.14) la exactitud del estimador LSE viene dada por:

$$C_{\theta\theta} = \alpha^2[\phi^T.\phi]^{-1} \qquad \text{(A.17)}$$

Desdichadamente, las propiedades expresadas por las ecuaciones (A.15) y (A.16) que deberían estar relacionadas con el ruido generalizado e(k), no llegan a ser tratadas por el estimador de mínimos cuadrados. La estructura del modelo paramétrico ilustrado por la figura A1, hace imposible obtener ruido blanco para e(k) lo que precisamos con la siguiente ecuación:

$$e(k) = A(q^{-1}).b(k) \qquad (A.18)$$

La característica de no-correlación expresada por la ecuación (A.15) no puede ser conocida por que e(k) está correlacionado con y(k) en la reac - ción dinámica del modelo, $A(q^{-1})$.

En resumen, el estimador LSE descrito, introduce errores sistemáticos en la estimación de parámetros. Para mejorar la calidad de las estimaciones otros métodos son intentados, por ejemplo el RLS que utiliza hasta tres valores sucesivos para la posición y. La generalidad del estimador de míni - mos cuadrados se basa en el siguiente principio: "Si e(k) es el ruido blanco y no esta correlacionado con $\Psi(k)$, entonces las propiedades anteriores se pueden conseguir agregando un filtro al ruido. Esto implica que deberán hacerse las consideraciones para dotar de cambios a la estructura del mo - delo paramétrico".

ANEXO B
OBTENCIÓN DE Jeff Y beff

ANEXO B
OBTENCIÓN DE Jeff Y beff

Las ecuaciones (2.20), (2.21) y (2.22) pueden rescribirse como:

$$Jeff\,\dot{\omega} = \left(\frac{KtKac}{R}\right)u - \left(beff + \frac{nEKt}{R}\right)\omega - f(n\omega) \qquad (B.1)$$

La función de transferencia lineal del sistema es derivada de la ecuación (B1) de la siguiente forma:

$$Jeff\,\dot{\omega} = \left(\frac{KtKac}{R}\right)u - \left(beff + \frac{nEKt}{R}\right)\omega \qquad (B.2)$$

Donde, utilizando la Transformada Laplace:

$$\omega(s) = \frac{\frac{nKtKac}{RJeff}}{s + \left(\frac{beff}{Jeff} + \frac{nEKt}{RJeff}\right)}u(s) \qquad (B.3)$$

La ganancia de continua se encuentra haciendo s=o:

$$\frac{\omega(0)}{u(s)} = \frac{nKtKac}{beffR + nEKt} = 467.32 radseg^{-1}V^{-1} \qquad (B.4)$$

Como el polo del sistema aparece en ω=47.0428radseg^{-1} y Kac=14.9 de la Tabla 1.2, se deriva lo siguiente:

$$\frac{beff}{Jeff} + \frac{nEKt}{RJeff} = \omega_{3db} = 47.0428 \qquad (B.5)$$

De las ecuaciones (B.3) y (B4) se obtiene: beff = 5.533x10^{-5} Nm/rads

Jeff = 5.6331x10^{-5} Kgm^2 que son los valores usados en la tabla 1.2.

ANEXO C
LISTADO DE LOS PROGRAMAS

ANEXO C
LISTADO DE LOS PROGRAMAS

Tendremos dos programas uno en lenguaje C++ llamado comp.cpp y el otro el equivalente en lenguaje ensamblador del Dsp56002 el sec.asm.

comp.cpp

```
#include "r1llib.h"
#include <stdio.h>
#include <math.h>
#include <assert.h>
#include <alloc.h>
#include <conio.h>
const float ESCALA= 2.*M_PI/(512.*19.7);
const float LOW_r = 1.*M_PI/4.;
const float HIGH_r= 0.;
/* Declaración de las funciones GORDAS */
void  Inicializa_Sistema();
void  Mide_Variables(float *yk, float *r, float *vel, float t);
void  Encuentra_Control(float r, float *u);
float Aplica_Control(float u, float ykp);
/* Declaración de las miles de variables globales que van a haber */
float a1, a2, b1, b2, U0, Cc;
```

```
float    Td;
float    t, tsimul;
float    Fs;
int      p0, pk;
/* Identificación */
float Thk[5], Th[5];
float ym1, ym2, um1, um2, ek, div, rt;
float Pk[5][5], Nk[5][5], Sk[5], Psk[5][5], Psn[5][5], Psaux[5][5];
float Phi[5], Phin[5], Phins[5], Ro;
float Cmax, Cmin, Co, LF;
float calculaC(); /* max(eig(Psk))/min(eig(Psk)) */
/* Control Optimo y Observador Optimo */
float y, x[2], z, TOL, rr, u;
float Q[3][3], R, Ge[3][3],He[3],Ce[3],K[3];
float Qb[2][2],Rb,G[2][2], H[2], C[2], Kb[2];
float P[3][3], Pb[2][2], Maux1[3][3], Maux2[3][3], Maux3[3][3];
float Vaux1[3], Vaux2[3], Vaux3[3], Vaux4[3], Raux1;
/* Aplicación del Control */
float VcFC, VcFE, Wth, Ofst;
/*********************************FUNCIONES*******************************/
float Aplica_Control(float u, float ykp)
{
        float v;
        if(fabs(ykp)<=Wth)
```

```
        {
                if(u<0.) v = -VcFE;
                if(u>0.) v =  VcFE;
        }
        else
        {
                if(u<0.) v = -VcFC;
                if(u>0.) v =  VcFC;
        }
        if(u+v<0.0)      u= 0.0;
        else if(u+v>1.4) u=  1.4;
        EnviarVoltage(u+v+Ofst);
        return u;
}
void Inicializa_Sistema()
{
        a1=-1.6246;          a2= .6246;
        b1=  .0479;          b2= .0410;
        /* Atención: Los valores de  a1, a2, b1, b2 dependen de la Fs */
        VcFE= .2;            VcFC= .15;
        Wth = .1;            Ofst= 2.3;
        Fs= 100.0;           U0  = 0.;
        Td= 1./Fs;           Cc  = 0.;
        t = 0.;              tsimul= 20.;
```

```
        G[0][0] = 0.; G[0][1]= 1.;              H[0] = 0.; C[0] = b2;
        G[1][0] =-a2; G[1][1]=-a1;              H[1] = 1.; C[1] = b1;
        Ge[0][0]= 0.;Ge[0][1]= 1.; Ge[0][2]=0.; He[0]= 0.; Ce[0]= b2;
        Ge[1][0]=-a2;Ge[1][1]=-a1; Ge[1][2]=0.; He[1]= 1.; Ce[1]= b1;
        Ge[2][0]=-b2;Ge[2][1]=-b1; Ge[2][2]=1.; He[2]= 0.; Ce[2]= 0.;
        P[0][0]= 0.1185;P[0][1]=- 0.0388;P[0][2]=- 0.0540;Q[0][0]= .1;Q[0][1]=
0.;Q[0][2]= 0.;
        P[1][0]=- 0.0338;P[1][1]= 0.32990;P[1][2]=- 0.1947;Q[1][0]= 0.;Q[1][1]=
.1;Q[1][2]= 0.;
        P[2][0]=- 0.0540;P[2][1]=- 0.1947;P[2][2]= 1.8086;Q[2][0]= 0.;Q[2][1]=
0.;Q[2][2]= .1;
        R =.05;K[0]=-0.5211;K[1]=1.3371;K[2]=-0.5137;
        Pb[0][0]=0.5386;Pb[0][1]=0.5605;Qb[0][0]=.01;Qb[0][1]=0.;
        Pb[1][0]=0.5605;Pb[1][1]=0.5986;Qb[1][0]=0.;Qb[1][1]=.01;
        Rb      =.04;Kb[0]=1.100;Kb[1]=1.160;
        y= 0.; x[0]= 0.; x[1]= 0.; z= 0.; y= 0.; u= 0.; TOL= 1.e-3;
        ym1=0.;
        ConfigurarHardware(0x272, 0x270, 0x271, 0x264, Fs);
        EnviarVoltage(Ofst);
        p0= LeerPosicion();
}
void Mide_Variables(float *yk, float *r, float *vel, float t)
{
        int i, j, k, n=2, niter;
```

```
double delta;
/* En primer lugar, la consigna: r= r(t)*/
if(t<=1.0) *r= 0.35+0.435398*t;
/* Ahora leo la posici'on */
pk= LeerPosición();
*yk = ym1 + ((float)(abs(pk-p0)>8000?0:pk-p0))*ESCALA;
p0= pk;
ym1= *yk;
/*restricción de la salida*/
if(*yk>*r)       *yk= 1.0*(*r);
else if(*yk<0.0)       *yk=0.0;
/* Y la salida */
y= *yk - *r;
/* Y la velocidad angular */
*vel= (5.*(*yk-ym1)+ *vel)/(1.+5.*Td);
/* Y vamos con el Observador: Primero, Ricatti */
/*Los valores de PB se obtienen de dlqe*/
/* Fin de encontrar la de Riccati */
/* Encontramos ahora la de Observacion */
/* Kb se puede obtener tambien de dlqe */
/* Y ahora actualizamos los estados */
rr= y - C[0]*x[0] - C[1]*x[1];
x[0]= x[0] + Kb[0]*rr;
x[1]= x[1] + Kb[1]*rr;
```

```
        z = z - y;
}
void Encuentra_Control(float r,float *U)
{
        /*Los valores de P de ricatti se envontraron con dlqr*/
        /* Fin de encontrar la de Riccati */
        /* Encontramos ahora la de control */
        /* Tambien pudieron hallarse con dlqr*/
        /* Ahora encontramos el control y el nuevo estado del Observador*/
        u= -(K[0]*x[0] + K[1]*x[1] + K[2]*z);
        Raux1= G[0][0]*x[0] + G[0][1]*x[1] + H[0]*u;
        x[1] = G[1][0]*x[0] + G[1][1]*x[1] + H[1]*u;
        x[0] = Raux1;
        *U = u;
}
void main()
{
        int ant=0, act=0, k, n;
        FILE *out;
        float *yv, *uv, *rv, vel, y, u, r;
        Inicializa_Sistema();
        clrscr();
        assert(tsimul<40);
        t= tsimul/Td;
```

```
n= (int)t;
assert(NULL!=(uv= (float *)calloc(n/5,sizeof(float))));
assert(NULL!=(yv= (float *)calloc(n/5,sizeof(float))));
assert(NULL!=(rv= (float *)calloc(n/5,sizeof(float))));
t= u= vel= y= 0.;
for(k=0; k<n;)
{
        act=NivelClock();
        if ((ant==0)&&(act==1))
        {
                Mide_Variables(&y, &r, &vel, t);
                Encuentra_Control(r, &u);
                u= Aplica_Control(u, vel);
                if(!(k%5))
                {
                        yv[k/5]= y;          rv[k/5]= r;          uv[k/5]= u;
                }
                k++;
                t += Td;       gotoxy(5,5); printf("%f",t);
                if(kbhit())    if(getch()==27) break;
        }
        ant=act;
}
EnviarVoltage(Ofst);
```

```
        out=fopen("comp.out","wt");
        for (int i=0; i<n/5; i++)
                fprintf(out,"%10f %10f %10f %10f %10f\n",
                                i*5*Td, rv[i], yv[i], uv[i], rv[i]-yv[i]);
        fclose(out);
}
```

sec.asm

```
        include 'ycomp1.asm'
tcsr    equ     $ffde
tcr     equ     $ffdf
pbc     equ     $ffe0
pbddr   equ     $ffe2
pbd     equ     $ffe4
pll     equ     $fffd
bcr     equ     $fffe
ipr     equ     $ffff
start   equ     $900
        org     y:$40
        dc      $000000,$000000,$000000
        org     y:$100
        dc      -0.5179
        dc      0.13460
        dc      -0.5128
        org     y:$114
        dc      0.9998
        dc      0.9999
        org     y:$400
        dc      $000000
        org     y:$1900
        dc      $000000
```

```
        org     P:$00
        jmp     $0040
        org     P:$0040
        movep   #$00,x:bcr
        movep   #$000000,x:pbc
        movep   #$261009,x:pll
        movep   #$000fff,x:pbddr
        movep   #$000012,x:tcsr
        movec   #$0300,sr
        movep   #$010000,x:ipr
        andi    #$cf,mr
        movep   #>$f4240,x:tcr
        move    #$900,r3
        move    #$1900,r5
in_1    jsr     xm1
        move    y:(r3)+,x0
        move    #$a90,a
        move    r3,y0
        sub     y0,a
        jset    #00,sr,*
        move    x0,a
        neg     a
        move    a,x0
        move    #.01,y0
```

```
        mpy     x0,y0,a
        lsl     a
        asr     a
        move    a,y:>$0043
        move    #$40,r1
        clr     b
        move    y:(r1)+,y0
        move    #$053f7d,x0
        mac     x0,y0,b y:(r1)+,y0
        move    #.0479,x0
        mac     x0,y0,b
        neg     b
        move    b,x0
        add     x0,a
        move    a,y:>$0044
x_0     move    y:>$0040,a
        move    y:>$0044,x0
        move    #.11,y0
        clr     b
        mac     x0,y0,b
        move    b1,x0
        move    #>$00a,y0
        mpy     x0,y0,b
        asr     b
```

```
        move    b0,x0
        add     x0,a
        move    a,y:>$0040
x_1     move    y:>$0041,a
        move    y:>$044,x0
        move    #.116,y0
        clr     b
        mac     x0,y0,b
        move    b1,x0
        move     #>$00a,y0
        mpy     x0,y0,b
        asr     b
        move    b0,x0
        add     x0,a
        move    a,y:>$0041
z       move    y:>$042,a
        move    y:>$043,y0
        sub     y0,a
        move    a,y:>$042
u       clr     a
        move    y:$40,x0
        move    #-0.5179,y0
        mac     x0,y0,a
        move    y:$42,x0
```

```
move    #-.5128,y0
mac     x0,y0,a
move    y:$41,x0
move    #>$00a,y0
mpy     x0,y0,b
asr     b
move    #>$113a93,x0
move    b0,y0
mac     x0,y0,a
neg     a
move    a1,y:$45
move    a,x0
move    #>$064,y0
mpy     x0,y0,b
asr     b
abs     b
move    b0,x0
move    y:$400,a
move    a1,a0
inc     a
move    a0,a1
move    a,y:$400
move    #>$015,y0
sub     y0,a
```

```
        jclr    #00,sr,apu
        move    #.2,b
        add     x0,b
        move    b,x0
apu     mpy     x0,#16,a
        move    a,y:(r5)
        move    y:$41,x0
        move    x0,y:$300
        move    #-.6246,y0
        move    y:$40,x0
        clr     a
        mac     x0,y0,a
        move    y:$41,x0
        move    #>$00a,y0
        mpy     x0,y0,b
        asr     b
        move    b0,y0
        move    #0.16246,x0
        mac     x0,y0,a
        move    y:$45,x0
        add     x0,a
        move    a1,y:$41
        move    y:$300,
        move    x0,y:$40
```

```
        jclr    #07,x:tcsr,*
        jmp     in_1
xm1     movep   y:(r5),x:pbd
        bset    #$00,x:tcsr
        rts
```

BIBLIOGRAFÍA

[1] Lab:PC+User Manual. Technical Report, National Instruments Corporation, 1994.

[2] Ceballos, Francisco Javier. Curso de Programación C++.

[3] IEEE August 1994 Vol.82, No8. Microprocesor an Digital IC'S for motion control.

[4] Isermann, Rolf. Digital Control Systems. Espringer-Verlag, 1991.

[5] Motorola INC. Manual de la familia DSP56000.

[6] Motorola INC. Manual del usuario DSP56002.

[7] Pascal, Renard. Implementation of Adaptive Controllers on the Motorola DSP56000/DSP56001[5]. Motorola INC, 1995.

[8] Viassolo, Daniel. Implementation of Digital Controllers. PhD dissertation, Purdue University, August 1996.

[9] Alan J. Laub, John N. Little, Clay M. Thompson y Andrew Grace. Control System Toolbox. The Maths Works Inc, 1992.

[10] Gene Moriarty; Farzad Nekoogar. Digital Control Using Digital Signal Processing. Prentice Hall PTR, 1999.

[11] John B. Moore, Brian Anderson. Optimal Control Linear Quadratic Methods. Prentice Hall International, 1989.

[12] Luis Lecina, Arturo Rojas. "Aplications of a Self-Tuning Controller to Nonlinear Processed Exposed to Nonlinear Disturbances," (1999).

[13] Stanley Lemeshow, David Hosner. "Applied Logistic Regression". John Wiley and Sons INC, 1989.

[14] Ogatha, Katsuiko. "Linear control System with Matlab" Prentice Hall,1994.

[15] Leonhard, Werner. "Control of electrical Drives" Springer, 1997.

[16] Isidori, Alberto. "Nonlinear Control Systems An Introduction" 2nd. Edition, Springer-Verlag, 1989.

[17] "The Student Edition of Simulink Dynamic System Simulation Software for Technical Education". The MATHWORKS Inc.