ISSN:1692-7257 Volumen 1 No 3 - 2004

CONTROLADOR FUZZY INDUSTRIAL

ING. ALEXANDER ARIAS LONDOÑO
ING. JUAN DIEGO LEMOS

Universidad de Antioquia,
Ciudad Universitaria, Bloque de Ingeniería,
aarias@udea.edu.co
jdlemos@universia.net.co
Medellín

ABSTRACT:
En este artículo se muestra el desarrollo e implementación de un controlador industrial mediante algoritmos de lógica difusa. El objetivo es el de construir un dispositivo que realice las labores de control en procesos industriales, compitiendo con los controladores PID. El desarrollo se hizo a partir del microcontrolador MC68HC912B32 de Motorola, cuya CPU soporta instrucciones de lógica difusa. El algoritmo de control estará programado en el microcontrolador, al igual que un conjunto de instrucciones de monitoreo y control a través de una conexión serial. El prototipo presenta una simplicidad en el manejo para evitar confusiones y problemas con operarios inexpertos.

KEYWORDS:
Microcontroladores, automatización, control, fuzzy logic.

.

1. INTRODUCCIÓN

El controlador más conocido y utilizado es el PID, el cual demostró su efectividad desde sus inicios; debido a esto se le han desarrollado mejoras para aumentar su aceptación dentro del mercado industrial; dentro de sus cambios esta el lazo adaptativo, que busca suplir su mayor desventaja, la necesidad de calcular sus parámetros de funcionamiento (constantes de acción proporcional, integral y derivativa). Recientemente se han implementado algoritmos de control inteligente (fuzzy, neurofuzzy, redes neuronales, algoritmos genéticos, etc.), con muy buenos resultados en diferentes campos, lo cual ha desviado la mirada hacia esta "nueva" alternativa de control. A diferencia de los PID, los controladores inteligentes no necesitan de parámetros de funcionamiento y obtienen resultados similares o talvez mejores que los PID.

2. DISEÑO DEL HARDWARE

El corazón del hardware del controlador es la tarjeta del microcontrolador de Motorola MC68HC912B32, desarrollada por Propuesta Dinámica, a partir de ella se implementó el Controlador; donde se encuentra la componente de software del diseño y en su periferia los demás componentes necesarios para realizar las labores concernientes al control.

El diagrama de bloques del figura 1 muestra la estructura general del controlador y la interconexión de todas las partes que lo conforman. Se puede observar que el microcontrolador es la parte central, y a su alrededor se presentan los demás elementos como son: la fuente de Alimentación de energía para el sistema, los módulos de visualización tanto el Display de Cristal Liquido como el panel de LEDs, el módulo de comunicación con el computador y el sistema de entrada/salida (I/O).

2.1 Interface de entrada/salida

Es la parte encargada de tomar la señal de entrada, y ejercer la acción control sobre la planta. Esta interface esta compuesta en su etapa de salida por un conversor Digital/Análogo (DAC0808), el cual es alimentado por las salidas del controlador correspondientes al puerto B (PORTB) del microcontrolador, su salida en corriente es acondicionada para obtener una salida en voltaje en el rango de 0 a 10v.

El diseño se realizó como sigue: - Se toma el valor recomendado para la resistencia en la entrada del DAC Vref+ (R3), que es de 5K para un voltaje de 12v, se selecciono un valor comercial de 5.6K con el mismo voltaje. La

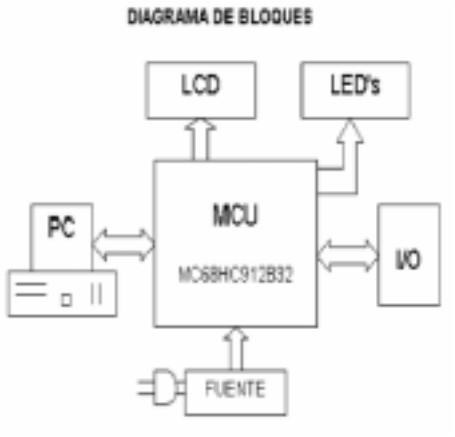

Fig. 1. Diagrama de Bloques

corriente máxima de salida (National Semiconductor, 1999) se presenta cuando todos los bits de entrada del DAC estén en alto, esto produce una corriente máxima de 2.14mA, como se observa en la siguiente

ECUACIÓN:

$$Iout = \frac{Vref}{R3}\left(\frac{A_1}{2}+\frac{A_2}{4}+\frac{A_3}{8}+\frac{A_4}{16}+\frac{A_5}{32}+\frac{A_6}{64}+\frac{A_7}{128}+\frac{A_8}{256}\right)$$

- El amplificador tiene un voltaje de salida dado por: Vout=R1xIout. Para obtener un voltaje máximo de 10v se calcula una resistencia de 4.7K

- Los demás valores de resistencias y condensadores utilizados son los recomendados por la hoja de datos del fabricante(National Semiconductor, 1999).

- El amplificador (Figura 2) tiene un voltaje de salida dado por: Vout=R1xIout. Para obtener un voltaje máximo de 10v se calcula una resistencia de 4.7K. -

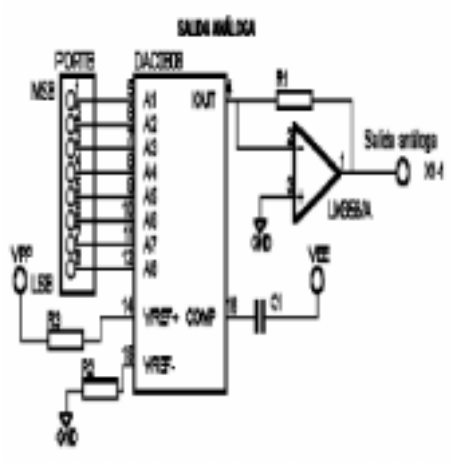

Fig. 2. Salida Análoga

Los demás valores de resistencias y condensadores utilizados son los recomendados por la hoja de datos del fabricante.

- Se puede seleccionar la salida de control en modo de PWM, correspondiente al bit 0 del puerto P (PP0) del microcontrolador (Motorola, 1999), cuyo ciclo de dureza es el mismo valor digital de la acción de control para brindar una alternativa de control. - El PWM es amplificado (Grafico 3) en magnitud por medio de un optoacoplador, el cual lo que hace es tomar el pulso de 5v que sale del microcontrolador y lo acopla a la fuente de 12v; esto hace que el PWM esté a disposición del usuario para su utilización. La entrada al controlador es una señal analógica con un rango entre 0-10v, la cual es acondicionada (Figura 4) para ser suministrada al bit 3 (ADDR03) del conversor Análogo/Digital del microcontrolador; dicho acondicionamiento consta de un divisor de voltaje para reducir su magnitud a 0-5v, luego un buffer con un Amplificador Operacional para evitar que choques de corriente pasen directamente al microcontrolador y por último un par de diodos para impedir que el valor del voltaje salga del rango que soporta el Conversor A/D (Motorola, 1999), los cuales entran en conducción cuando el valor se sale del rango.

los componentes necesarios para la interface De entrada/salida son los siguientes:
- U1 - Conversor D/A - DAC0808.
- U2 - Amplificador operacional doble - LM358N.
- UK1 - Optoacoplador - 4N35.
- D1, D2 - Diodos - 1N4148.
- C1 - Condensador - 0.1µF/50v.
- R1, R2, R4, R5, R7 - Resistencias - 4.7 KΩ.
- R3 - Resistencia - 5.6 KΩ.
- R6 - Resistencia - 680 Ω.

2.2 Etapas de visualización

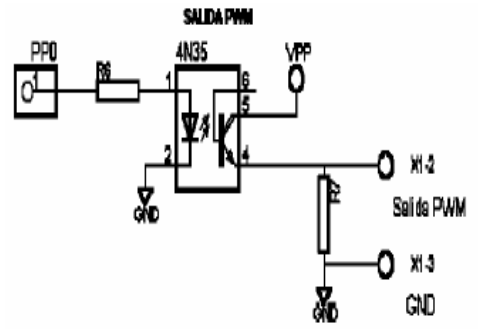

Fig. 3. Salida PWM

La visualización de la información se realiza de tres formas: por medio de la comunicación serial con un computador, a través de un Display de Cristal Liquido de 2 filas x 16 columnas, y una barra de LEDs.

La visualización en el LCD se realiza de modo

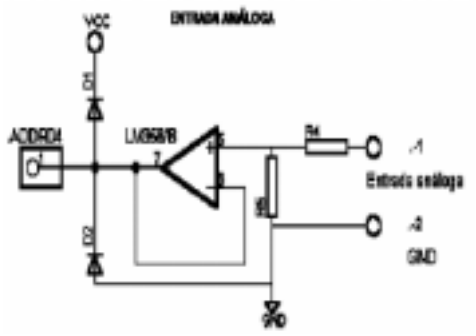

Fig. 4. Entrada Análoga

constante y es controlado por seis líneas, dos de control y cuatro de datos (configuración a 4 bits), las cuales corresponden a seis bits del puerto A (PORTA), además cuenta con un potenciómetro para controlar el contraste de la visualización.

2.3 Comunicación con el computador

La interface serial es el estándar RS232 en configuración Modem-Null, la cual consiste en realizar la comunicación serial sin protocolo, para lo cual fue necesario cortocircuitar las líneas de control de la comunicación en el terminal DB 9, como son RTS con CTS (7 y 8) y DCD con DTR y DSR (1,4 y 6) (Valvano, 2000); además se debe configurar la sesión de HiperTerminal en el computador a 9600 bps, 8 bits de datos, 1 bit de parada, ninguna paridad y ningún control de flujo. El diagrama esquemático se muestra en la figura 5.

Los componentes requeridos para el Ensamble de la interface RS232 son los siguientes:

- IC1 - Chip de interface RS232 - MAX232.
- X1 - Conector DB 9 hembra - F09 HF.
- C1, C2, C3, C4, C5 - Condensadores - 10 µF/25v.

2.4 Teclado de control

El teclado de control del sistema consta de 2 pulsadores conectados directamente a dos pines del puerto T (PT7 y PT6), con los cuales se incrementa y decrementa el Set Point del controlador, e inmediatamente el

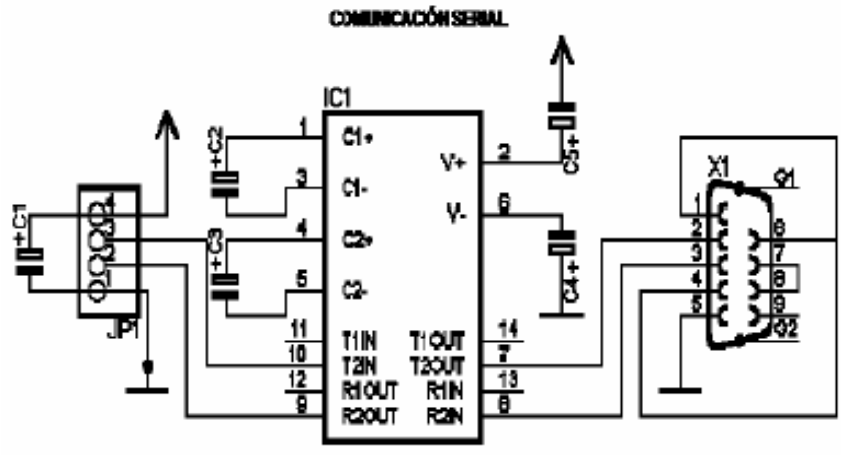

Fig. 5. Comunicación Serial

microcontrolador lo actualiza y realiza el procedimiento de control.

3. DISEÑO DEL SOFTWARE

La componente de software del controlador es la programación del microcontrolador, la cual se hizo a través del compilador en C para HC12 ICC12 para Windows versión 5. El diagrama de flujo general del programa en el microcontrolador se muestra en la Figura 6

Los procesos de inicialización de puertos, interrupciones y periféricos se refieren a las instrucciones que permiten definir el comportamiento de los puertos como entrada o salida. Las interrupciones se habilitan para realizar todas las labores dispuestas en el controlador y los periféricos (como el LCD) que requieren de un proceso de programación antes de ser utilizado para la labor que cumple. En el modo de espera, el microcontrolador entra en un modo de bajo consumo de energía en los momentos en que ninguna labor se esté realizando, solo sale de él cuando se presente una interrupción en el microcontrolador. Las tareas son actividades específicas dentro del esquema del controlador; como actividades se tienen las siguientes:

- Desarrollo del proceso de control Fuzzy.
- Atención al cambio de Set Point por los pulsadores.
- Actualización de los datos en el LCD.
- Comunicación serial con el PC.
- Actualización de la barra de LEDs

Fig. 6. Diagrama de Flujo

Fig. 6. Diagrama de Flujo

3.1 Inicialización

Inicialización de Puertos

El microcontrolador tiene registros donde se configura el comportamiento de los puertos, estos son llamados registros de control. Los bits de control se pueden programar con ceros o unos, haciendo que los pines del puerto sean de entrada o salida respectivamente. Los puertos utilizados son:

- Seis líneas del puerto A para controlar la visualización en el display de cristal liquido) (Valvano, 2000) (PA7 con RS, PA6 con E, PA5 con D4, PA4 con D5, PA3 con D6 y PA2 con D7).
- Todo el puerto B, el cual es la salida digital hacia el conversor digital/análogo (PB7 con A1, PB6 con A2... y PB0 con A8).
- Cinco líneas del puerto DLC, para el control de la barra de LEDs (PDLC2 a PDLC6).

Inicialización de Periféricos

- Serial La comunicación serial esta configurada a 9600 baudios, 8 bits de datos,

ninguna paridad y un bit de parada.
- Análogo a Digital Se adquieren datos por el canal 4 (ADDR04), con conversión secuencial a 8 bits.
- Timer de muestreo
Está configurado como Output Compare, con preescalado de 8; lo cual combinado con el reloj del sistema (8Mhz), genera un tic de 1µs, para ser utilizado en el programa principal en la administración de tareas.
- Display de Cristal Liquido
El display es configurado a cuatro bits, ellas se encuentran en el puerto A.
- Teclas de incremento y decremento
Se configuran los Timers 6 y 7 como Input Capture, con ello se capturan los flancos de subida; con la posibilidad de dejar presionada la tecla para obtener el mismo efecto en forma continua.
- PWM Se pone a funcionar el PWM, con una frecuencia de 10 Khz., alineado a la izquierda y ciclo de dureza iniciado en cero.

3.2 Realización de tareas Proceso de Control Fuzzy

El proceso es realizado en varias etapas, para mayor claridad se explican las etapas y luego se ponen las rutinas:
• Un timer de muestreo el cual despierta el proceso, cada 200µs, activando la etapa de adquisición de datos en el ATD.
• Conversión Analógica a Digital, la cual entrega el resultado del Pen Point en un rango de 0 a 255; para luego activar la rutina Fuzzy.
• Rutina de control Fuzzy, esta toma los datos del Set Point y el Pen Point para entregar la acción de control. El tiempo de muestreo está acorde con el tiempo que tarda la rutina Fuzzy en realizarse, el cual se rige por la siguiente
relación: NC = 37S + 3R + 47; donde NC es el número de ciclos, S es el número de conjuntos difusos y R es el número de reglas; con lo anterior, para 11 conjuntos difusos y 121 reglas, para un máximo de 817 ciclos, y como el reloj del sistema es de 8MHz, cada ciclo tarda 0.125µs, por la tanto la rutina gasta 103µs aproximadamente.

• Actualización del dato de control, con el valor entregado por la rutina Fuzzy, se renuevan los datos del conversor Digital a Análogo (PORTB) y el ciclo de dureza del PWM (PP0). Después de realizado lo anterior se activan las banderas de atención a interrupciones y tareas.

Atención a teclas de control
Las teclas funcionan con una sola pulsación o dejándolas presionadas. Las interrupciones por captura de entrada se utilizan para incremento y decremento. Cuando se dejan presionadas, el cambio se realiza cada 100ms.

Visualización
Esta se compone de un Display de Cristal Liquido,donde se muestran estandarizados de 0 a 100%:
AC: Acción de Control.
EA: Error Absoluto.
SP: Set Point.
PP: Pen Point.
También incluyen unos LEDs de visualización del error absoluto, los cuales van de 0 a 10%.

Comunicación con el computador
Esta se encarga de la transmisión y la recepción por serial, en ella se tienen algunos comandos disponibles en la recepción, pueden ser en mayúsculas o minúsculas y son de un solo carácter, ellos son:
• H ó h: Muestra una ayuda en línea.
• S ó s: Muestra el Set Point, Pen Point, Error

Absoluto y Acción de Control.
• I ó i: Incrementa Set Point.
• D ó d: Decrementa Set Point.
C ó c: Limpia pantalla de Hiper Terminal.

Modo de Espera
El microcontrolador se pone en un modo de bajo consumo hasta que una interrupción lo despierte, luego continua realizando las tareas consecutivamente.

Funciones generales
Son algunas funciones utilizadas en las rutinas previamente explicadas, ellas realizan conversiones de números a cadenas de caracteres, y también cambios de rango.

3.3 conjuntos y reglas del controlador fuzzy

El diseño de los conjuntos y las reglas se basa en un conocimiento empírico sobre el sistema que se quiere controlar. En este trabajo se desarrollan unos conjuntos y reglas que pueden ser típicos para sistemas simples, si se quiere se pueden cambiar al criterio de cada diseñador (Zadeh, 1965). La estructura de control se muestra en el Figura 7

Conjuntos Difusos de las Variables de Entrada
Set Point**:** En el momento de montar el control en la práctica se suelen usar nombres, los conjuntos difusos se muestran en la figura 8. Para evitar confusiones entre variables, el Set Point se llama in1, se definirán sus conjuntos como
Bajo → in1_0
Medio → in1_1
Alto → in1_2
Pen Point: Se llama in2, sus conjuntos serán (Figura
9):
Bajo → in2_0

Fig. 7. Estructura de Control

Medio → in2_1
Alto → in2_2
Conjuntos Difusos de la variable de salida
En la práctica se uso un microcontrolador MC68HC912B32, el cual trabaja con instrucciones difusas y para realizar los cálculos utiliza funciones de salida singleton (Kosko, 1997), por comodidad se trabajará sobre lo que se espera directamente.

La salida de control (Figura 10) se podría llamar out,
y sus conjuntos:
Bajo → out_0
Medio → out_1
Alto → out_2

Reglas que Gobiernan el Sistema
Estas reglas se sacaron de forma que no haya cambios bruscos y el sistema se comporte de manera óptima. if Set Point es Bajo y Pen Point es Bajo entonces Salida de Control es Baja if Set Point es Bajo y Pen Point es Medio entonces Salida de Control es Baja if Set Point es Bajo y Pen Point es Alto entonces Salida de Control es Media if Set Point es Medio y Pen Point es Bajo entonces Salida de Control es Media if Set Point es Medio y Pen Point es Medio entonces Salida de Control es Media if Set Point es Medio y Pen Point es Alto

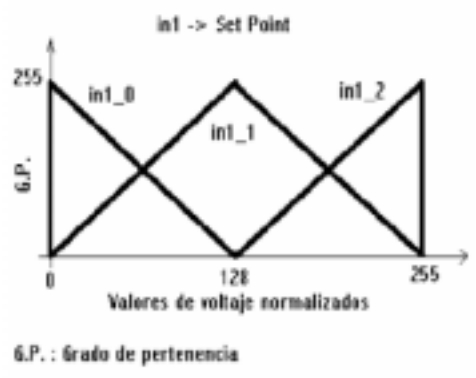

Fig. 8. Set Point

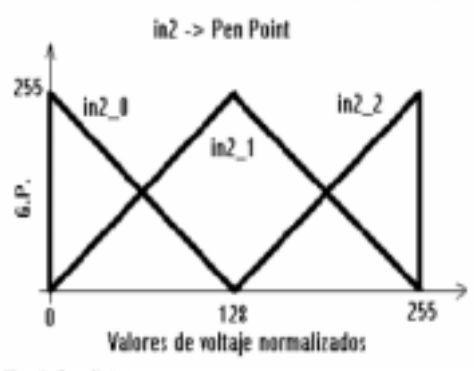

Fig. 9. Pen Point

entonces Salida de Control es Media if Set Point es Alto y Pen Point es Bajo entonces Salida de Control es Media if Set Point es Alto y Pen Point es Medio entonces Salida de Control es Alta if Set Point es Alto y Pen Point es Alto entonces Salida de Control es Alta

Con una matriz de reglas se da una mirada más clara del comportamiento de las reglas del sistema, la matriz se presenta a continuación:

La salida de control está dada por las que están en cursiva y negrilla.

La forma en que se programan los conjuntos

Fig. 10. Salida de Control

y reglas en el microcontrolador es el siguiente (Motorola, 1997):

//Fuzzy

Para el anterior conjunto de reglas se desarrolló una función en lenguaje ensamblador para poder aprovechar las instrucciones de lógica difusa que soporta el microcontrolador (Motorola, 1997) y fue creada con el fin de que el código principal

Pen Point Set Point	Bajo	Medio	Alto
Bajo	***Baja***	***Baja***	***Media***
Medio	***Media***	***Media***	***Media***
Alto	***Media***	***Alta***	***Alta***

Matriz de Reglas

en C pudiera hacer uso de ella como la función FuRuDe() (Van, 1994),

el código es el que sigue:

```
#define FUZZY_SETS     3
#define  In1_0    0
#define  In1_1    1
#define  In1_2    2
#define  In2_0    3
#define  In2_1    4
#define  In2_2    5
#define  Out_0    6
#define  Out_1    7
#define  Out_2    8
//Variables en RAM
unsigned char v_Fuz_InyOut[3*FUZZY_SETS];
unsigned char v_Set_Point=0;
unsigned char v_N_Set=0;
unsigned char v_Pen_Point=0;
unsigned char v_Error =0;
unsigned char v_Control=0;
//Valores en Flash: tablas, constantes
const unsigned char
Fuz_Inputs_MFs[]={
                    //Entrada uno,Set Point
                    0x00,0x80,0x00,0x02,
                            0x00,0xFF,0x02,0x02,
                            0x80,0xFF,0x02,0x00,

                    //Entrada 2, Pen Point
                            0x00,0x80,0x00,0x02,
                            0x00,0xFF,0x02,0x02,
                            0x80,0xFF,0x02,0x00
                            };

const unsigned char
Fuz_Out_MFs[]={
                    //Salida, Accion de Control
                            0x00,0x80,0xFF };

const unsigned char
Fuz_Reglas[ ]={
        In1_0,In2_0,0xFE,Out_0,0xFE,
        In1_0,In2_1,0xFE,Out_0,0xFE,
        In1_0,In2_2,0xFE,Out_1,0xFE,

        In1_1,In2_0,0xFE,Out_1,0xFE,
        In1_1,In2_1,0xFE,Out_1,0xFE,
        In1_1,In2_2,0xFE,Out_1,0xFE,

        In1_2,In2_0,0xFE,Out_1,0xFE,
        In1_2,In2_1,0xFE,Out_2,0xFE,
        In1_2,In2_2,0xFE,Out_2,0xFF     };
```

4. CONCLUSIONES

1. Es posible controlar por medio de técnicas difusascualquier proceso industrial, con la mayor facilidad y con el mínimo de errores, basta con conocer su comportamiento general para estructurar una serie de conjuntos difusos y sus respectivas reglas.

2. Los algoritmos de control difuso presentan

```
_FuRuDe::
Fuzzifier::
        ldx     #_Fuz_Inputs_MFs
        ldy     #_v_Fuz_InyOut
        ldaa    _v_Set_Point
        ldab    #3   ;FUZZY_SETS
In1_Loop::
        mem
        dbne    b,In1_Loop
        ldaa    _v_Pen_Point
        ldab    #3   ;FUZZY_SETS
In2_Loop::
        mem
        dbne    b,In2_Loop
        ldab    #3   ;FUZZY_SETS
Rule_Eval::
        clr     1,y+
        dbne    b,Rule_Eval
        ldx     #_Fuz_Reglas
        ldy     #_v_Fuz_InyOut
        ldaa    #$FF

        rev
Defuzzifier::
        Ldy     #_v_Fuz_InyOut+6 ;v_Fuz_Out
        ldx     #_Fuz_Out_MFs
        ldab    #3   ;FUZZY_SETS
        wav
        ediv
        tfr     y,d
        stab    _v_Control
        rts
```

la ventaja de funcionar sin estar variando parámetros de funcionamiento, pero es posible que para diferentes plantas sea necesario crear otro sistema de conjuntos y reglas que brinden mayor precisión.

3. El desarrollo del controlador con un microcontrolador que soporta instrucciones de lógica difusa, facilita las labores de control, ya que el tiempo de procesamiento de reglas es mucho más eficiente y rápido.

4. El sistema fue probado con plantas de segundo y cuarto orden, con las cuales funcionó bastante bien, presentando un bajo error absoluto en estado estable y rapidez en la respuesta sin sobreimpulsos grandes.

REFERENCIAS BIBLIOGRÁFICAS

(1). (Kosko, 1997) Kosko, B. *Fuzzy Engineering* University of Southern California. 1997, 547 pages. ISBN 0133537315 - Prentice Hall.

(2) (Motorola, 1997) *68HC12*, *CPU12* Reference
Manual, Motorola, 1997.

(3) (Motorola, 1999) *MC68HC912B32 Advance Information*, rev 2.0, Motorola, 1999.
(National Semiconductor, 1999)]

(4) *National Semiconductor Database*, Analog and Interface products. National Semiconductor, 1999.

(5)(Valvano, 2000) Valvano, J. W. *Embedded Microcomputers Systems*, Real Time Interfacing, Pacific Grove,Brokes/Cole, 2000.

(6) Van, 1994) VAN , T. *Programming Microcontrollers in C*, San Diego, Technology Publishing, 1994.

(7)(Zadeh, 1965) Zadeh, L. A. *Fuzzy Sets*. Journal of Information and Control, 8: 338-353, 1965.