Informes Técnicos Ciemat

759
Junio 1995

Control de una garra robotizada mediante un controlador borroso

J. Alberdi
J.M. Barcala,
E. Gamero
J.J. Navarrete

Dirección de Tecnología

Toda correspondencia en relación con este trabajo debe dirigirse al Servicio de Información y Documentación, Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas, Ciudad Universitaria, 28040-MADRID, ESPAÑA.

Las solicitudes de ejemplares deben dirigirse a este mismo Servicio.

Los descriptores se han seleccionado del Thesauro del DOE para describir las materias que contiene este informe con vistas a su recuperación. La catalogación se ha hecho utilizando el documento DOE/TIC-4602 (Rev. 1) Descriptive Cataloguing On-Line, y la clasificación de acuerdo con el documento DOE/TIC.4584-R7 Subject Categories and Scope publicados por el Office of Scientific and Technical Information del Departamento de Energía de los Estados Unidos.

Depósito Legal: M-14226-1995
NIPO: 238-95-010-2
ISSN: 0214-087X

Editorial CIEMAT

CLASIFICACIÓN DOE Y DESCRIPTORES

420203, 990200
ROBOTS, MATERIALS HANDLING, FUZZY LOGIC, MATERIALS HANDLING EQUIPMENT, ELECTRONIC CIRCUITS, ELECTRONIC GUIDANCE.

"Control de una garra robotizada mediante un controlador borroso"

Alberdi, J.; Barcala, J.M.; Gamero,E.; Navarrete, J.J.;
38 pp. 8 figs.

Resumen

Una garra robotizada es controlada mediante un sistema basado en lógica borrosa. El sistema está basado en un microcontrolador borroso NLX230. Las reglas de control son programadas en la memoria del microcontrolador por un procesador 68020. El esfuerzo y su derivada sirven como señales de realimentación en el control. Este sistema puede adaptar su esfuerzo a la resitencia mecánica del objeto entre los dedos.

"Control of a mechanical gripper with a fuzzy controller"

Alberdi, J.; Barcala, J.M.; Gamero,E.; Navarrete, J.J.;
38 pp. 8 figs.

Abstract

A fuzzy logic system is used to control a mechanical gripper. System is based in a NLX230 fuzzy microcontroller. Control rules are programmed by a 68020 microprocessor in the microcontroller memory. Stress and its derived are used as feedback signals in the control. This system can adapt its effort to the mechanical resistance of the object between the fingers.

INDICE.

1. Introducción.

Los conjuntos borrosos son un concepto matemático propuesto por L.A. Zadeh en 1965. La lógica que opera sobre estos conjuntos, denominada lógica borrosa o difusa, es una generalización de la lógica multivaluada y contiene la lógica booleana clásica como caso particular.

Los ordenadores, cuyo funcionamiento se basa en la lógica booleana, operan sobre informaciones precisas, manejando enunciados que únicamente pueden ser ciertos o falsos. En la lógica clásica un elemento pertenece o no pertenece a un conjunto determinado, sin término medio posible. Por lo tanto un elemento no puede pertenecer al mismo tiempo a un conjunto y a su complementario. Esta construcción preserva la estructura de la lógica y evita que un objeto sea y no sea algo al mismo tiempo.

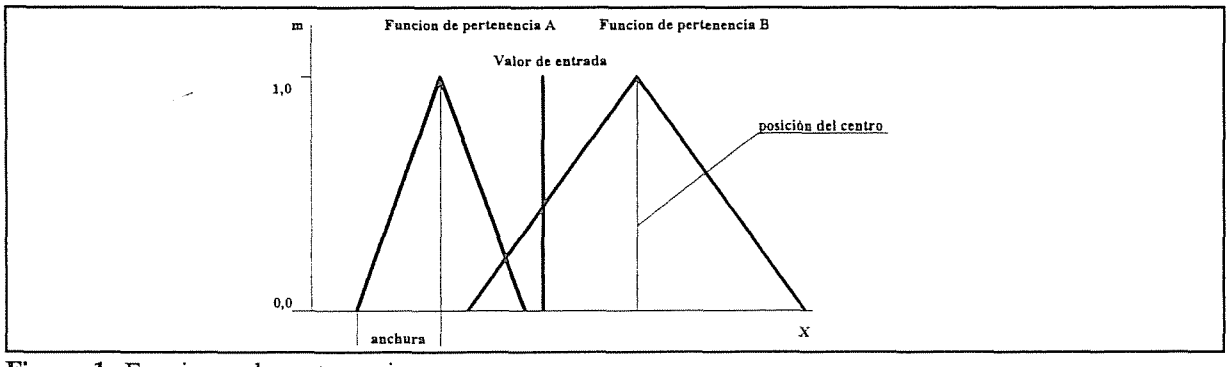


Figura 1. Funciones de pertenencia.

Sin embargo, los humanos manejamos habitualmente información con un alto grado de imprecisión. Conceptos como "alto", "bajo", "joven", "viejo", "lejos" o "cerca", por citar algunos, son usados a diario. Es difícil determinar que representan estas valoraciones exactamente. ¿ Qué altura es necesaria para ser alto ? ¿ 1,75 o 1,80 o quiza es suficiente con 1,70 ? Lo mismo sucede con la edad , ¿ cuántos años son necesarios para ser considerado viejo o joven ? No hay dudas en los casos extremos (a los 20 se es joven y a los 100 se es viejo), pero los problemas empiezan cuando hay que lidiar con valores intermedios y decidir donde se encuentra la frontera.

A pesar de estas imprecisiones todos somos capaces de comprender la información contenida en frases con este tipo de aseveraciones. De hecho prácticamente toda la información que recibimos o transmitimos a lo largo del día se intercambia mediante estos conceptos vagos e imprecisos, pero que tienen un significado real y útil para cualquier persona. Este tipo de "lógica del sentido común" es el que se identifica con la lógica borrosa.

La lógica borrosa maneja estos conceptos vagos, como "demasiado deprisa" o "poco caliente", a los que se denomina etiquetas linguísticas.

En la lógica borrosa un elemento puede pertenecer en parte a un conjunto. Mientras que los conjuntos tradicionales tienen fronteras claramente definidas, las fronteras de los

conjuntos borrosos se diluyen. La curva que representa la frontera borrosa se denomina función de pertenencia. La falta de nitidez en los límites permite que un predicado pueda ser parcialmente cierto y parcialmente falso al mismo tiempo.

No debe entenderse esta falta de precisión como equivalente a aleatoriedad. El azar, manejado mediante la teoría de probabilidades, intenta establecer si un hecho sucederá o no en el futuro basándose en una amplia estadística recopilada en el pasado. La lógica borrosa trata sobre hechos que ya están sucediendo pero de los que se tiene una información insuficiente. El tratamiento que se hace de esta información no tiene nada que ver con probabilidades o aleatoriedad.

2. Teoría del control borroso.

Las técnicas del control borroso son conocidas desde 1975, pero no ha sido hasta los últimos años, con la aparición de múltiples aplicaciones exitosas, especialmente en Japón, cuando se ha generalizado el interés por este método de control.

Todos los métodos de control tradicionales se basan en dos puntos fundamentales: primero, el sistema a controlar debe ser conocido, es decir, puede predecirse la respuesta del sistema a cualquier entrada, y segundo, el parámetro a controlar debe poder expresarse en fórmulas matemáticas concisas como función de las variables del sistema. Métodos de control basadas en estos supuestos son los más extendidos hoy en día y prácticamente los únicos que se usan. Pero a medida que la complejidad del sistema aumenta los supuestos anteriores tienden a fallar debido a la aparición de efectos no lineales, al carácter no estacionario del sistema o a la falta de un conjunto de datos suficientemente representativo.

Por muy elaborados y sofisticados que sean estos controladores todos tropiezan con dificultades insalvables cuando enfrentan tareas como aparcar un coche o reconocer caras, tareas que los humanos realizan sin aparente esfuerzo.

Los controladores borrosos se construyen a base de reglas del tipo "if ..... then ....." ("Si *el esfuerzo aumenta* entonces *disminuye la fuerza*") que transforman una serie de conjuntos borrosos de entrada en otros de salida. Gracias a ellos la experiencia acumulada puede usarse como método de control.

Para ello se definen una serie de zonas en la variable de entrada que se solapan relacionadas mediante las reglas "if ..... then ....." que proporcionan una curva de salida formada a partir de las reglas activas en cada momento. Esta salida no es utilizable directamente por los controladores por lo que debe realizarse un proceso de "desemborronamiento" que transforme la salida en un valor numérico concreto.

Son muy numerosos los ejemplos de aplicaciones comerciales exitosas, casi todas japonesas, basadas en la lógica borrosa. Desde el control de un tren al autoenfoque de las cámaras, aspiradoras, automóviles, supervisión de bases de datos, gestión en bolsa, control de un horno de cemento y multitud de aparatos domésticos y productos electrónicos. Según

el Ministerio de Comercio Internacional e Industria de Japón en 1992 el mercado de productos basados en lógica borrosa movió 2000 millones de dólares.

3. Descripción de la garra robotizada.

Entre 1991 y 1993 el Ciemat, en colaboración con el Instituto de Automática Industrial del C.S.I.C., construyó un manipulador robotizado pensado para aplicaciones en procesos de ensamblado. En este manipulador o garra se implementó un método de control tradicional basado en un microcontrolador HCTL-1100 con un filtro PD y una interfase con un ordenador personal que permite al usuario fijar los parámetros de funcionamiento.

La garra consta de una estructura de aluminio en forma trapezoidal en cuyo lado más ancho se encuentran los husillos que permiten el movimiento de los dedos del manipulador.En la parte central de la estructura metálica se encuentra alojado el motor c.c. que acciona los engranajes que transmiten el par a los husillos. La parte más importante del manipulador son los dedos. Estos han sido diseñados para poder instalar puentes completos de galgas extensiométricas, que permitan, por un lado medir el esfuerzo que se está ejerciendo sobre la pieza manipulada y por otro detectar posibles choques con obstáculos imprevistos. Se han colocado tres puentes por dedo para medir el esfuerzo según los tres ejes de un sistema cartesiano de coordenadas. En el apéndice A pueden encontrarse varias fotografías de la garra.

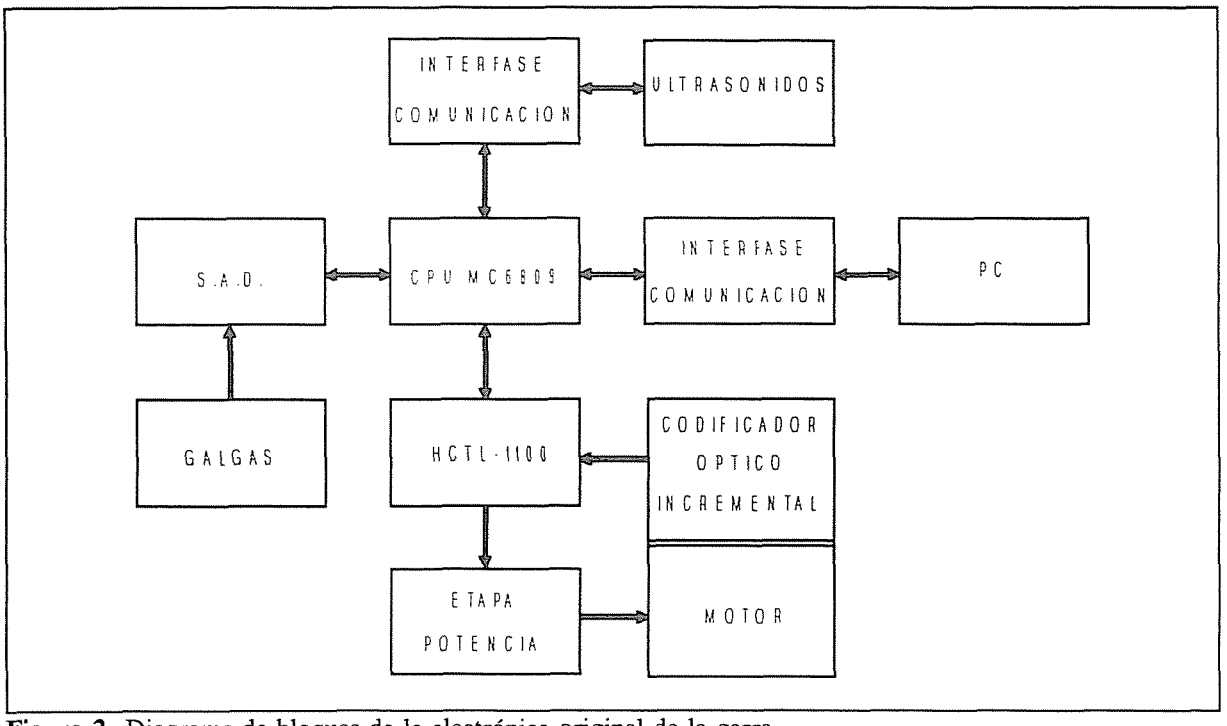


Figura 2. Diagrama de bloques de la electrónica original de la garra.

La electrónica de este sistema está distribuida en dos tarjetas. Una de ellas se monta en la estructura metálica de la garra y la otra ha sido diseñada para ser insertada en el bus de un PC. En la primera de ellas se encuentra la electrónica asociada a los puentes de galgas que miden los esfuerzos mecánicos de los dedos, a un sensor de ultrasonidos usado como detector de presencia y al suministro de potencia al motor. En la segunda se encuentra un microprocesador MC6809 y el microcontrolador HCTL-1100 encargado de dirigir el motor.

El programa de control del manipulador está distribuido en dos bloques principales: uno ejecutado por el MC6809 y otro ejecutado por el ordenador personal en el que se haya colocado la tarjeta de interfase.

El PC es una estación de comandos que permite al usuario un fácil manejo de las posibles tareas que puede realizar el manipulador. Entre otras tareas, este programa calcula los parámetros de control y monitoriza el esfuerzo de los dedos del manipulador. Además, es posible hacerle trabajar con o sin ayuda del sensor de ultrasonidos. Esto facilita la posibilidad de que sea el usuario quien de la orden de cerrar los dedos para coger la pieza o bien que sea el mismo programa de control el que realice esta misión.

El software de la interfase inteligente tiene como misión principal la ejecución de los comandos. Para realizar dichos comandos, el microprocesador de esta tarjeta, un MC6809, cuenta con la ayuda del controlador de motor HCTL-1100, gracias al cual, se descarga de una parte importante del trabajo. Cuenta también con la información que procede de los sensores de esfuerzo y del detector de presencia. La tarea realizada durante la fase control consiste en mantener el esfuerzo fijado por el usuario dentro de unos márgenes adecuados. Además puede detectar los posibles choques que ocurran entre la garra y los obstáculos que fueren encontrados durante la manipulación.

4. Control de una garra robotizada mediante un controlador borroso.

Como alternativa a este sistema de control, el grupo de Automática y Robótica del Ciemat desarrolló un nuevo procedimiento basado en lógica difusa con la intención de dotar de una mayor autonomía al controlador, mejorando su capacidad de decisión y su adaptabilidad ante diferentes sucesos.

La finalidad de este trabajo era controlar el funcionamiento de la garra ya descrita mediante técnicas borrosas de control. Se pretendía medir de alguna forma el contacto de la pinza con un objeto y modificar la presión con que los dedos sujetaban ese objeto en función de la resistencia que presentara. Así, los dedos no actuarían con la misma fuerza sobre un objeto blando que sobre otro duro y también cambiarían su comportamiento en función del peso del objeto. Igualmente se pretendía que el controlador pudiera reaccionar ante posibles deslizamientos o caídas de una pieza ya recogida.

Para ello se desarrolló una electrónica de control adecuada basada en el FMC (fuzzy microcontroller) NLX230 de NeuraLogix. Este microcontrolador de tecnología CMOS procesa hasta 30 millones de reglas por segundo, es fácil de configurar y necesita un mínimo de componentes externos. Asimismo se definió un conjunto de reglas que controlaban la

velocidad y el sentido de giro del motor en función del esfuerzo medido por las galgas extensiométricas y de su derivada.

4.1. El controlador borroso NLX230.

El FMC es un controlador especializado. En vez de usar algoritmos ejecutados secuencialmente para controlar una salida en función de las condiciones de entrada, actúa en paralelo, aplicando un conjunto de reglas a un vector de entradas. Estas reglas determinan que acciones se tomarán según las condiciones de entrada.

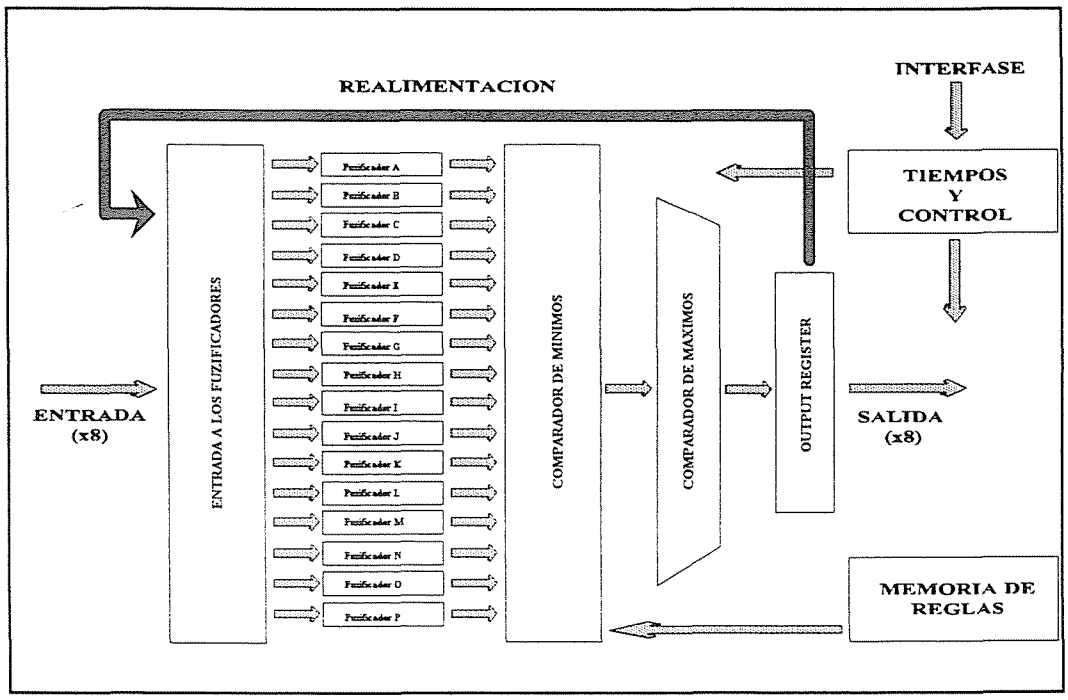


Figura 3. Diagrama de bloques del microcontrolador difuso NLX230.

Los dispositivos FMC intentan sustituir a los microcontroladores convencionales ofreciendo mayor eficacia y menores costes de implementación. Estos dispositivos emplean los principios de la lógica borrosa para calcular la salida óptima según las condiciones de entrada. Los valores de entrada son valorados en función de lo bien que se ajustan a un conjunto de funciones de pertenencia definidas por el usuario. Para conseguir una implementación digital eficiente el NLX230 usa funciones de pertenencia simétricas y lineales y el método de inferencia min/max más simple.

Un conjunto de reglas definidas por el programador permite determinar que condiciones se hallan presentes en las entradas. Cada regla consta de hasta dieciséis términos, uno para cada par entrada/función de pertenencia, y un Valor de Acción para

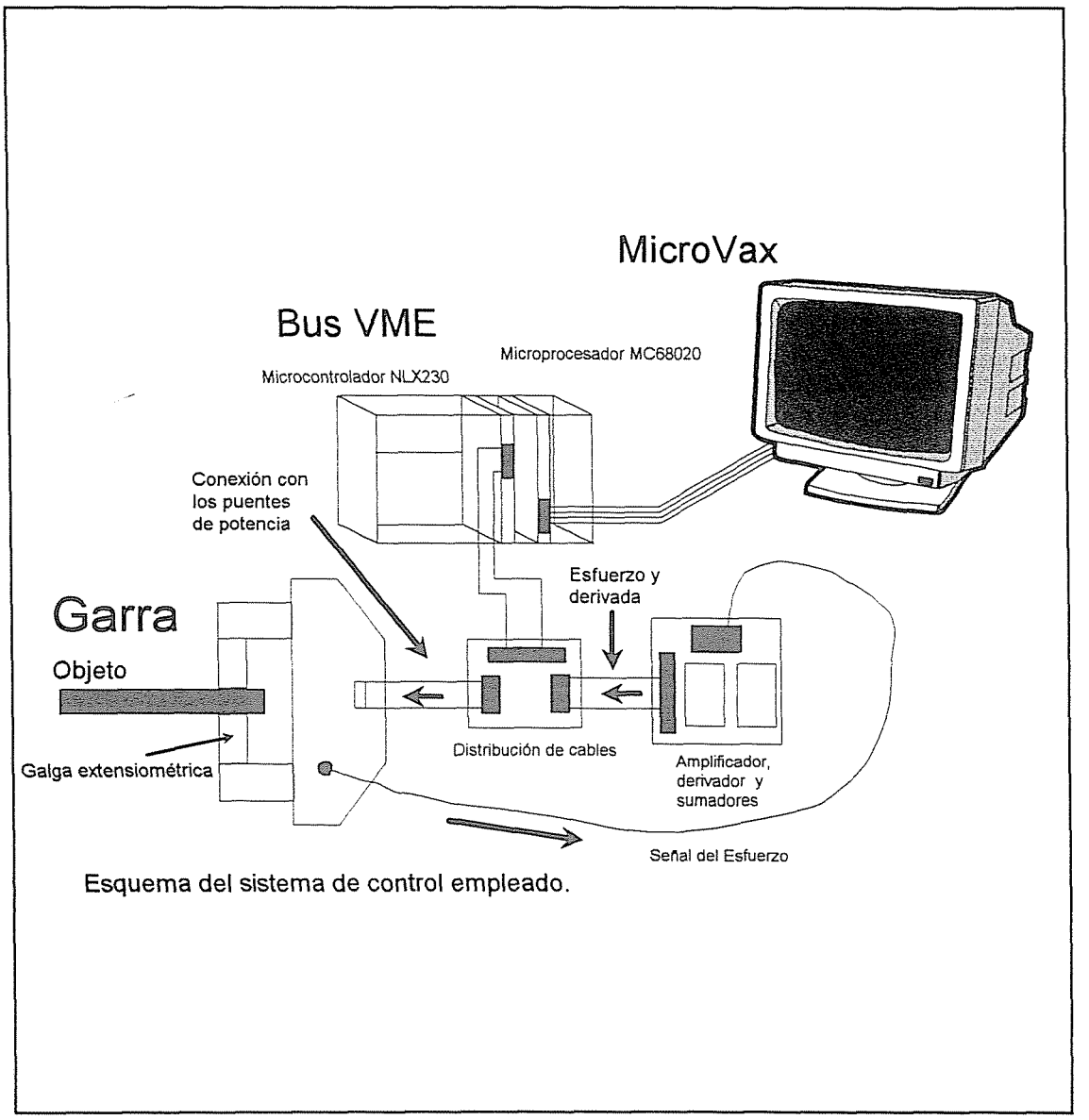
modificar la salida cuando se activa esa regla. La regla que mejor se adapta a las condiciones de entrada es la que determina como se modificará la salida.

El chip admite la definición de un máximo de 64 reglas, que se almacenan en registros de 24 bits. Cada salida puede programarse para usar tantas reglas como necesite (hasta un máximo de 64). El número de reglas disponible depende únicamente de las que ya estén siendo usadas por otras salidas.

El NLX230 acepta un máximo de ocho entradas digitales. Estas entradas pueden asociarse a cualquiera de los dieciséis "fuzzificadores" para formar los pares entrada/función de pertenencia.

Cada función de pertenencia está definida por dos valores: la posición de su centro y su anchura. Un "fuzzificador" calcula la distancia que hay entre el valor de entrada y el centro de la función de pertenencia. Después compara esta distancia con la anchura de la función. Si la distancia es menor que la anchura, la entrada forma parte de la función de pertenencia y la distancia se complementa para obtener el grado de pertenencia.

Para complementar se resta la distancia de la mayor distancia posible (31). Cuanto más cerca esté el valor de entrada del centro mayor es el grado de pertenencia. Aquellos valores diferentes de cero pasan al comparador de mínimos para el procesamiento de las reglas. Aquí se haya el valor mínimo para cada regla. Estos valores se pasan al comparador de máximo para obtener el mayor valor presente. Este valor determina cual es la regla ganadora y cual será la acción que se tomará como salida.



Esquema del sistema de control empleado.

Figura 4. Esquema del sistema de control.

4.2. Electrónica de control.

4.2.1. Descripción.

El soporte físico sobre el que se ha implementado el sistema de control está basado en el bus VME. Como CPU se ha utilizado una tarjeta MVME-133A de Motorola que tiene un procesador M-68020 de 32 bits. Para conectar el controlador borroso se ha desarrollado una tarjeta cuyo diagrama bloque puede verse en la figura 5. Sus esquemas completos pueden verse en los apéndices finales.

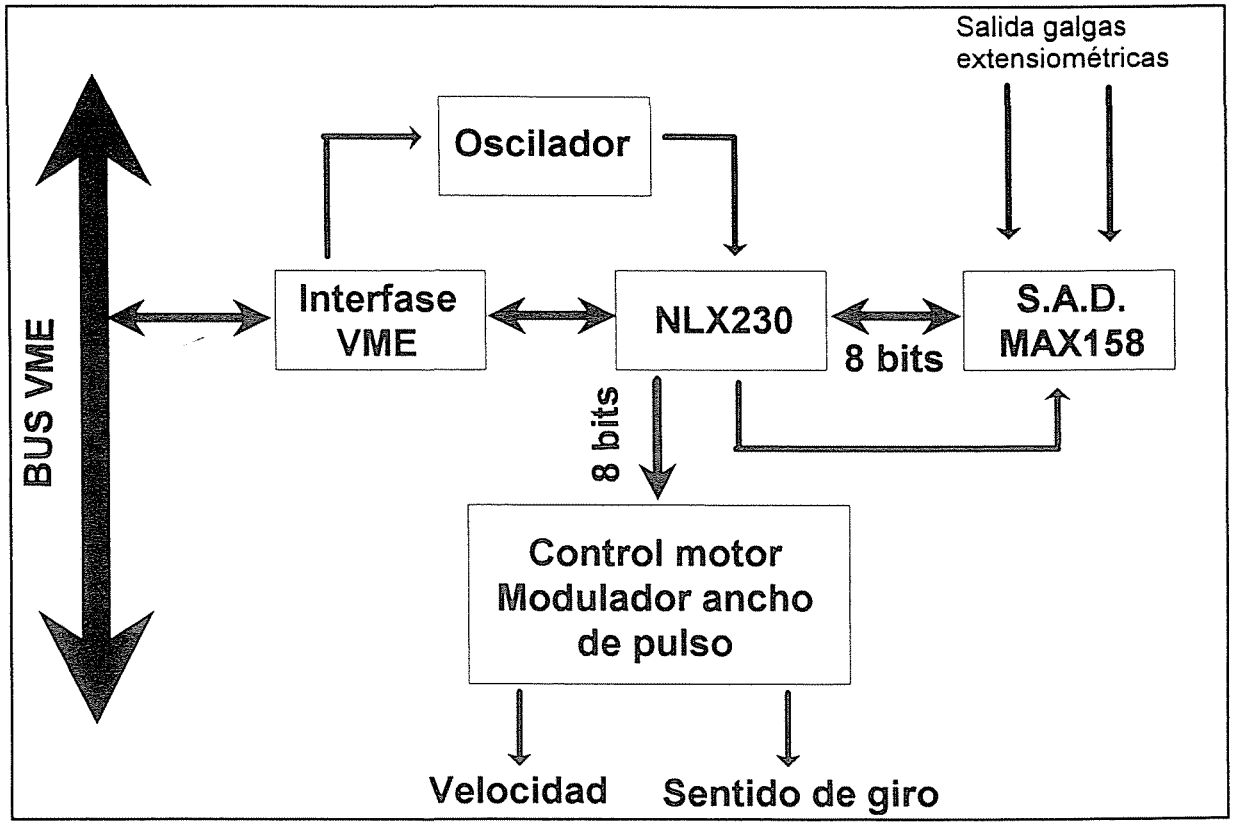


Figura 5. Diagrama de bloques de la tarjeta con el controlador borroso.

Desde el punto de vista del procesador digital el NLX230 está organizado como una memoria de 2048 bits. Internamente están organizados en 256 registros de 8 bits, pero la lectura/escritura de estos registros se realiza secuencialmente, bit a bit, a través de dos líneas DO (lectura) y DI (escritura). La CPU no tiene acceso directo a ninguno de los registros por lo que estos se cargan en bloque trasvasando íntegramente todo el contenido de un mapa de memoria.

La interfase VME garantiza la conexión de controlador difuso con el bus. Es del tipo esclavo y se compone de los amplificadores de línea, el decodificador de direcciones y un registro de control que permite fijar el modo de operación del controlador.

Los puentes de galgas extensiométricas proporcionan señales que son amplificadas y tratadas por una electrónica situada en la propia garra. Existen dos grupos de señales: las

proporcionales al esfuerzo y las proporcionales a la derivada del esfuerzo con relación al tiempo. Se sabe por lo tanto cómo es el esfuerzo y cómo varía con el tiempo. Estas señales, entre 0 y 5V, son leídas por el NLX230 a través de un sistema de adquisición MAX-158 que consta de un multiplexor de 8 canales y de un convertidor A/D de 8 bits de resolución.

La salida digital del controlador es recogida por un circuito modulador de anchura de pulso, cuya salida está compuesta por dos señales: un tren de impulsos de frecuencia fija pero de anchura proporcional a la salida del NLX-230, y un nivel digital 0/1. Estas dos señales gobiernan los circuitos de potencia del motor: el tren de impulsos determina la velocidad y el nivel el sentido de giro.

4.2.2. Modo de operación.

Los 256 registros internos del NLX230 contienen la información necesaria sobre el número y la localización de las variables de entrada, "fuzzificadores" asociados con cada una de las entradas y reglas a aplicar en el control.

En un sistema de desarollo basado en un PC las reglas se transforman en una tabla o mapa de memoria de los registros. La tabla se transmite via RS232 al 68020 que a su vez la carga en el controlador. En un futuro está previsto utilizar el Motorola en tareas de programación lo que permitiría modificar los parámetros de control de forma permanente y utilizar la garra en diversas tareas, simultáneamente. Finalizada la escritura de los registros el procesador da la orden de comenzar activando el correspondiente bit en el registro de control. A partir de este momento el NLX230 empieza a trabajar de forma autónoma. Sincronizado por un oscilador gobierna el sistema de adquisición de datos y con la información recibida elabora la señal de salida que transmite al control del motor. Para parar el sistema basta con modificar el registro de control.

4.3. Programación del controlador borroso.

La señal de esfuerzo que se usa como entrada proviene del puente de galgas extensiométricas señalado en la figura 6. Este puente es el más sensible a los esfuerzos en dirección perpendicular a la superficie de contacto de los dedos. Esta señal de esfuerzo también se derivaba para obtener la segunda señal de entrada usada en el control.

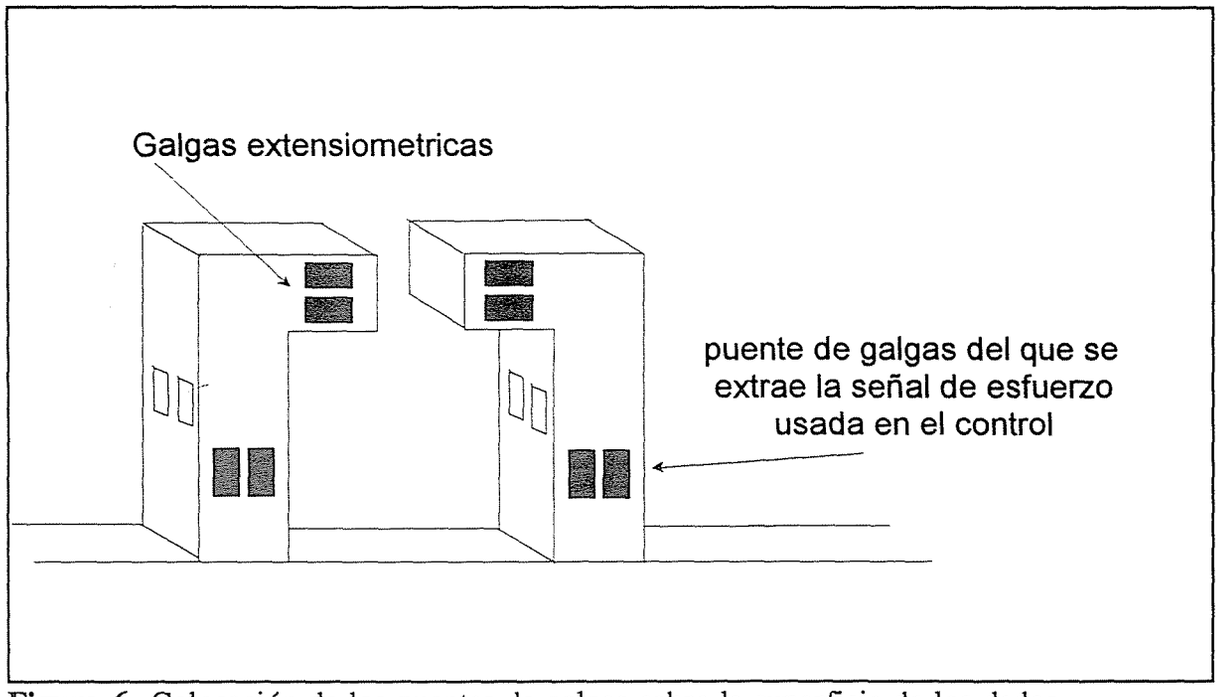


Figura 6. Colocación de los puentes de galgas sobre la superficie de los dedos.

Las dos variables de entrada (esfuerzo y su derivada) se dividieron de la manera siguiente:

Variable	*Etiqueta*	*Valor*	*Anchura*
Esfuerzo	Muy Poco	20	30
	Poco	60	30
	Medio	70	30
	Bastante	90	30
	Mucho	110	30
Derivada del esfuerzo	Muy Negativa	85	30
	Negativa	100	30
	Cero	120	30
	Positiva	140	30
	Muy Positiva	160	30

ya que el NLX230 solo usa funciones de pertenencia con la forma de un triángulo isósceles de altura unidad los datos de la posición del centro (valor) y su semibase (anchura) son suficientes para definir inequívocamente esas funciones.

El sistema de desarrollo ha permitido probar con diversos conjuntos de reglas y diferentes definiciones de términos. Después de varias pruebas el conjunto de reglas que parecía más eficaz era:

Tabla I

Reglas usadas en el controlador

Esfuerzo / Derivada	Muy Poco	Poco	Medio	Bastante	Mucho
Muy Negativa	Cierra	Cierra	Cierra	Cierra	Cierra
Negativa	Cierra	Cierra	Cierra Despacio	Cierra Despacio	Cierra Despacio
Cero	Abre Muy Rápido	Abre Muy Rápido	Abre Despacio	Para	Para
Positiva	Abre Rápido	Abre Rápido	Abre Despacio	Para	Para
Muy Positiva	Abre Despacio	Abre Despacio	Abre Despacio	Para	Para

Velocidades según el esfuerzo y su variación.

Las reglas recogidas en esta tabla deben leerse de la manera siguiente,

IF Esfuerzo **IS** Muy Poco
AND Derivada **IS** Muy Negativa
THEN Velocidad **IS** Cierra

Como el actuador final no puede comprender ordenes de este tipo sino que necesita instrucciones concretas, los valores de salida de esta tabla pueden traducirse de la manera siguiente :

Variable	Etiqueta	Valor
Velocidad	Cierra	-117
	Cierra Despacio	-112
	Para	0
	Abre Despacio	5
	Abre Rápido	20
	Abre Muy Rápido	40

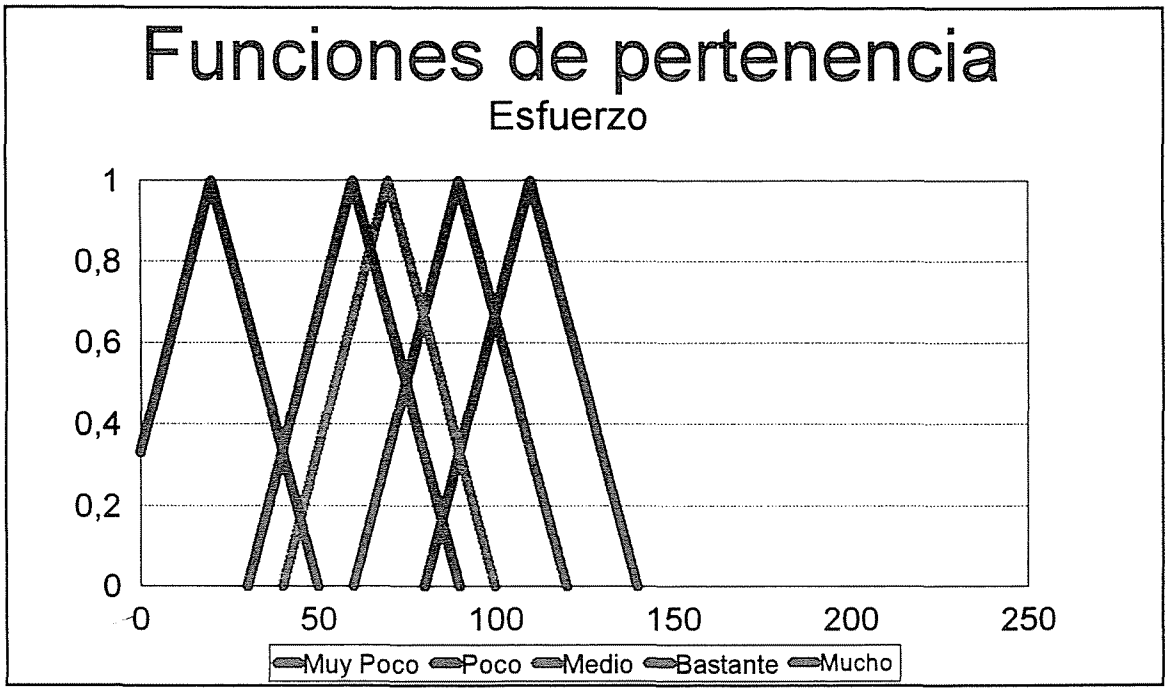


Figura 7.

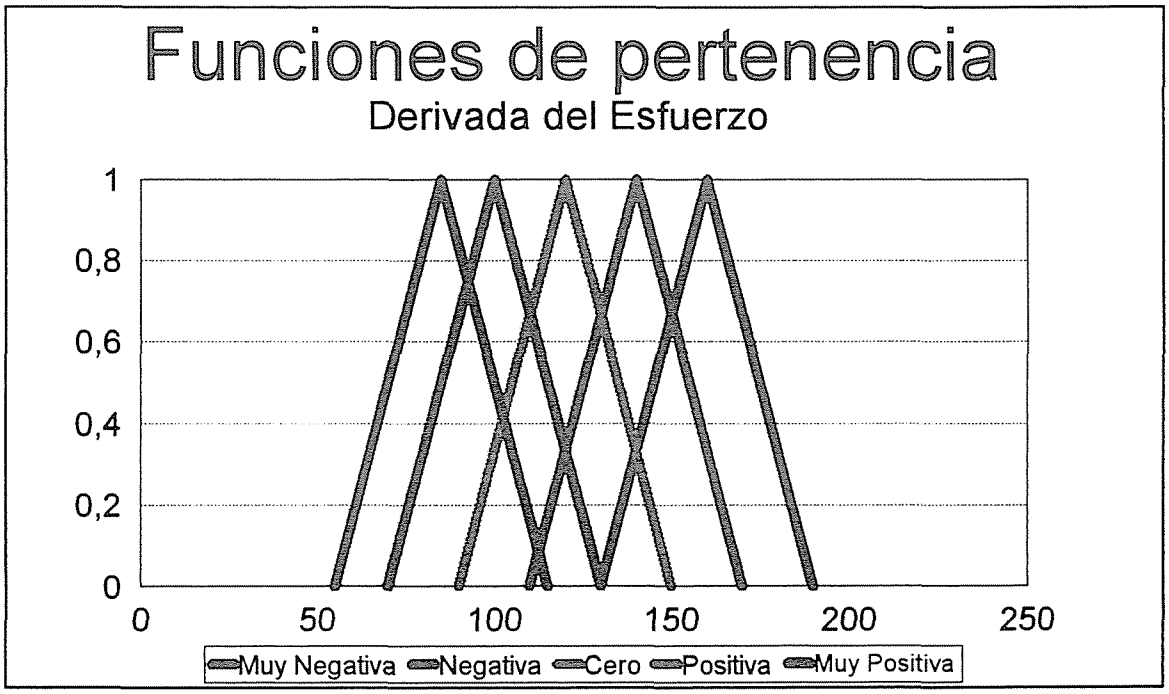


Figura 8.

5. Conclusiones.

Esta experiencia puso de manifiesto una de las cualidades ya conocidas de los sistemas de lógica difusa: facilidad de implementación, con el consiguiente ahorro de tiempo y medios. Además se consiguieron reacciones suaves y más progresivas del controlador con lo que disminuyeron los cabeceos del motor provocados por los cambios constantes del controlador tradicional.

Durante las pruebas del sistema la garra probó su destreza con diversos objetos como palillos, bolígrafos, gomas de borrar, cajas de varios tamaños, alicates y algunas otras herramientas. Nunca rompió ninguno de ellos pero podía mantenerlos con firmeza. La eficacia con la que mantenía el esfuerzo variaba según la geometría de la zona con la que contactaba. Si el objeto tenía caras lisas el controlador mantenía el esfuerzo sin vibraciones pero si contactaba con una arista tenía más dificultades para fijar el esfuerzo y aparecían algunos cabeceos.

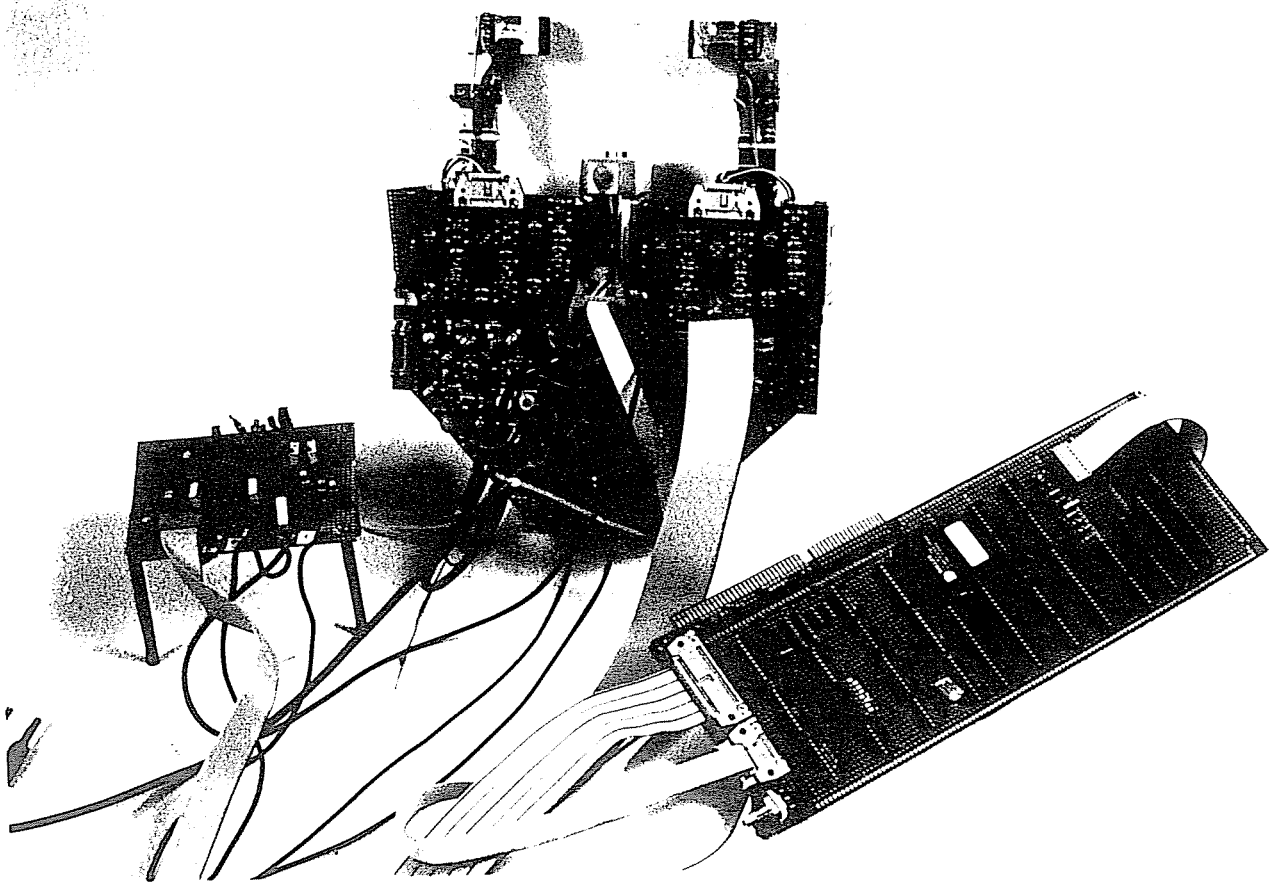
Sin embargo no se advirtieron diferencias de comportamiento debidas a la rigidez del objeto a coger. El controlador mostró la misma eficacia tanto para cuerpos duros como blandos, demostrando una buena adaptación a cada caso.

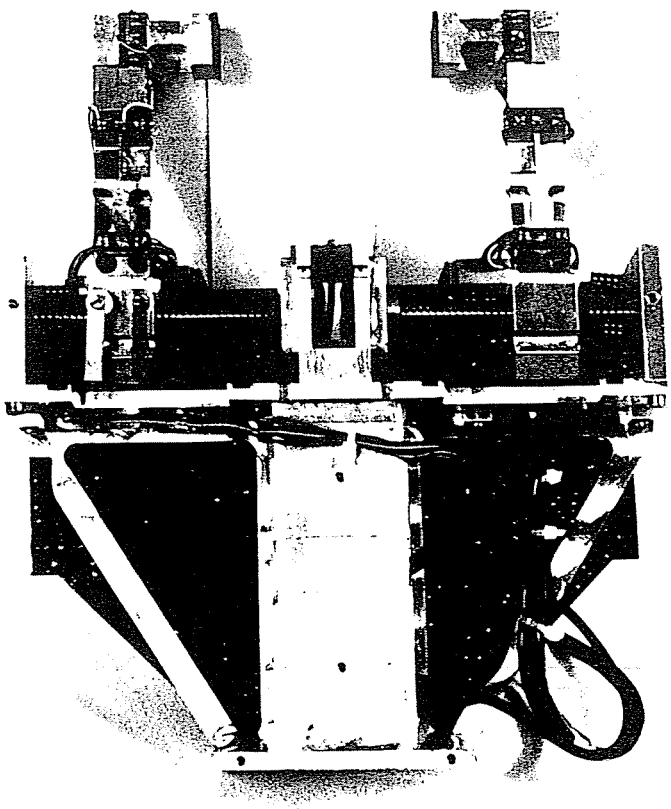
Un problema que se encontró fue la dependencia de la salida del microcontrolador de sus entradas en ese instante. Esto, que en general es una virtud deseable del sistema, hacía que la respuesta de la garra cuando el objeto se rompía o se caía no fuera la adecuada. Para solucionar este problema se recurrió a un pequeño cambio en la electrónica de control con el fin de conseguir que la conducta del controlador fuera más duradera en el tiempo. Para ello se añadió un flip-flop en la línea de la señal de sentido de giro. Este flip-flop se activa por la señal de controlador y es borrado por los finales de carrera de la garra.

Después de la modificación el controlador, cuando reconocía que le retiraban el objeto que tenía entre sus dedos o cuando ese objeto se rompía, era capaz de abrir la garra para separar los dedos y prepararse para recoger otro elemento.

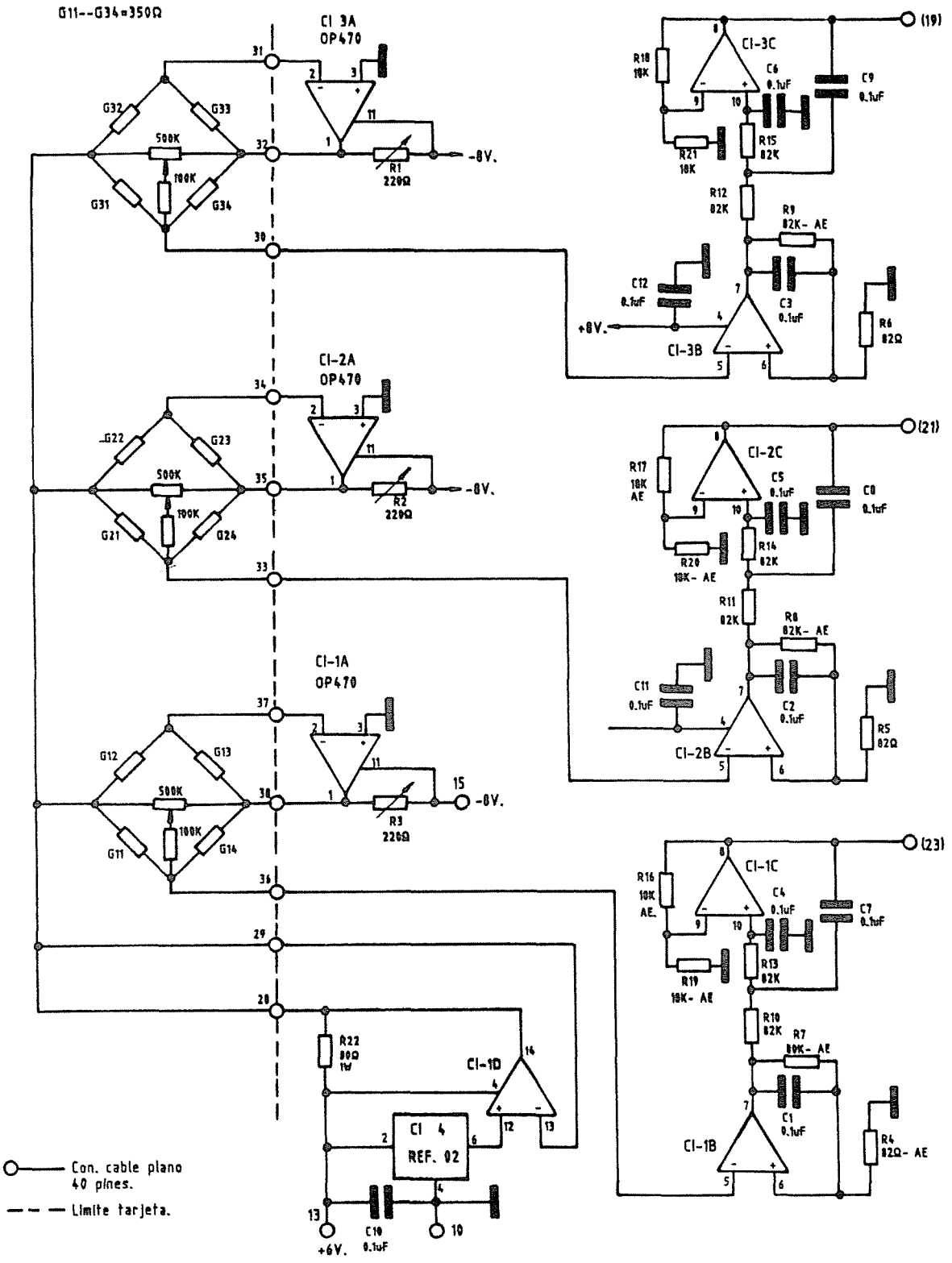
El principal inconveniente del sistema es que el controlador no puede asegurarse de que tiene un objeto entre los dedos. Como la única referencia del exterior es el esfuerzo en los dedos, la garra puede cerrarse hasta que sus dedos entren en contacto uno con otro y entonces pararse y mantener el esfuerzo como si realmente hubiera agarrado otro cuerpo. Esto limita el comportamiento autónomo de la garra y obliga a que sea el operario el que dé la orden de ejecución. Esta dificultad puede subsanarse usando un sensor de presencia de ultrasonidos, tal y como se empleó en el primer diseño de la garra.

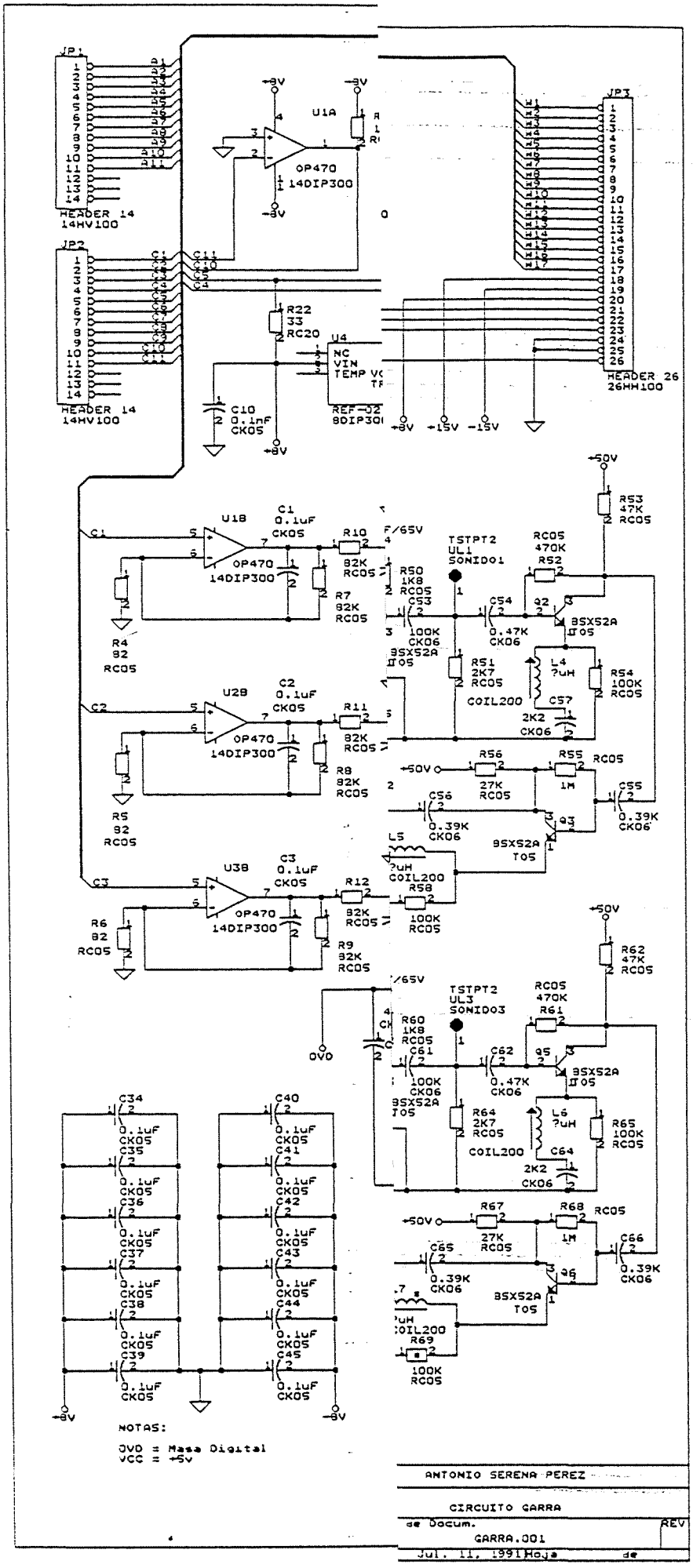
Apéndice A. Fotografías de la garra.

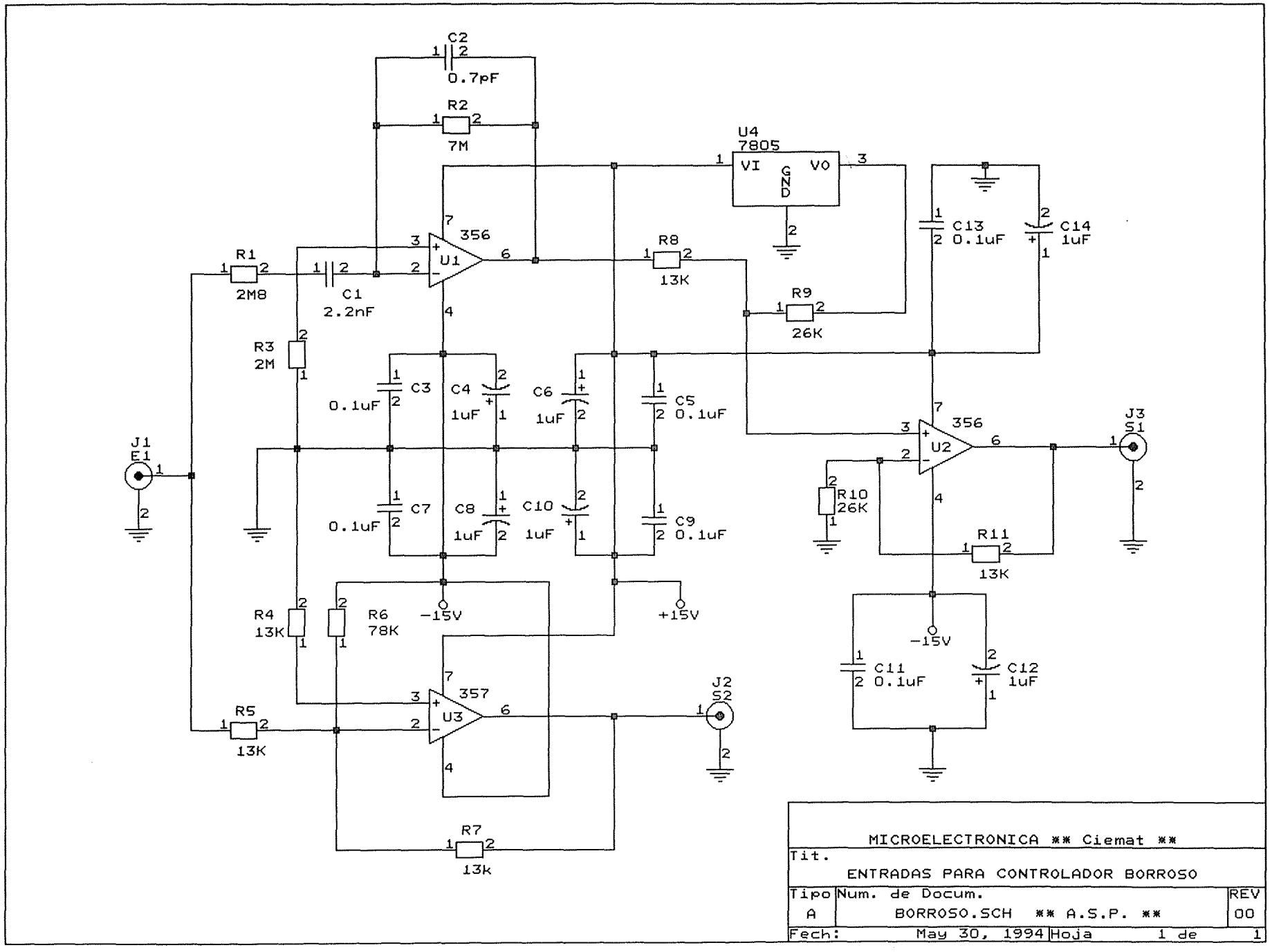


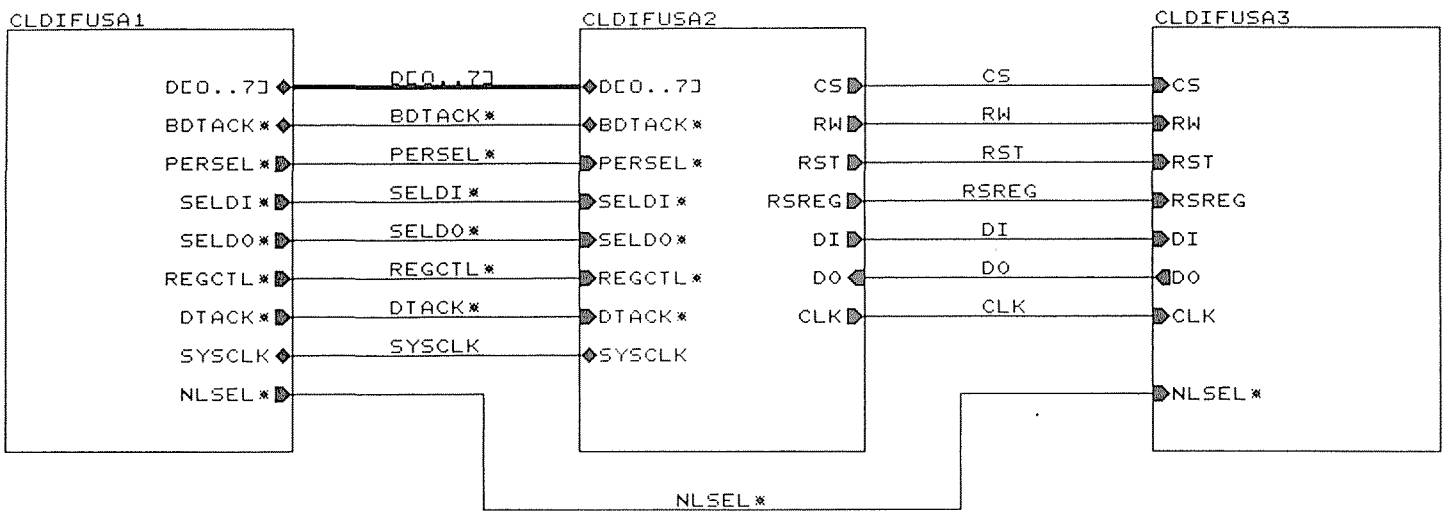


Apéndice B. Esquemas electrónicos.

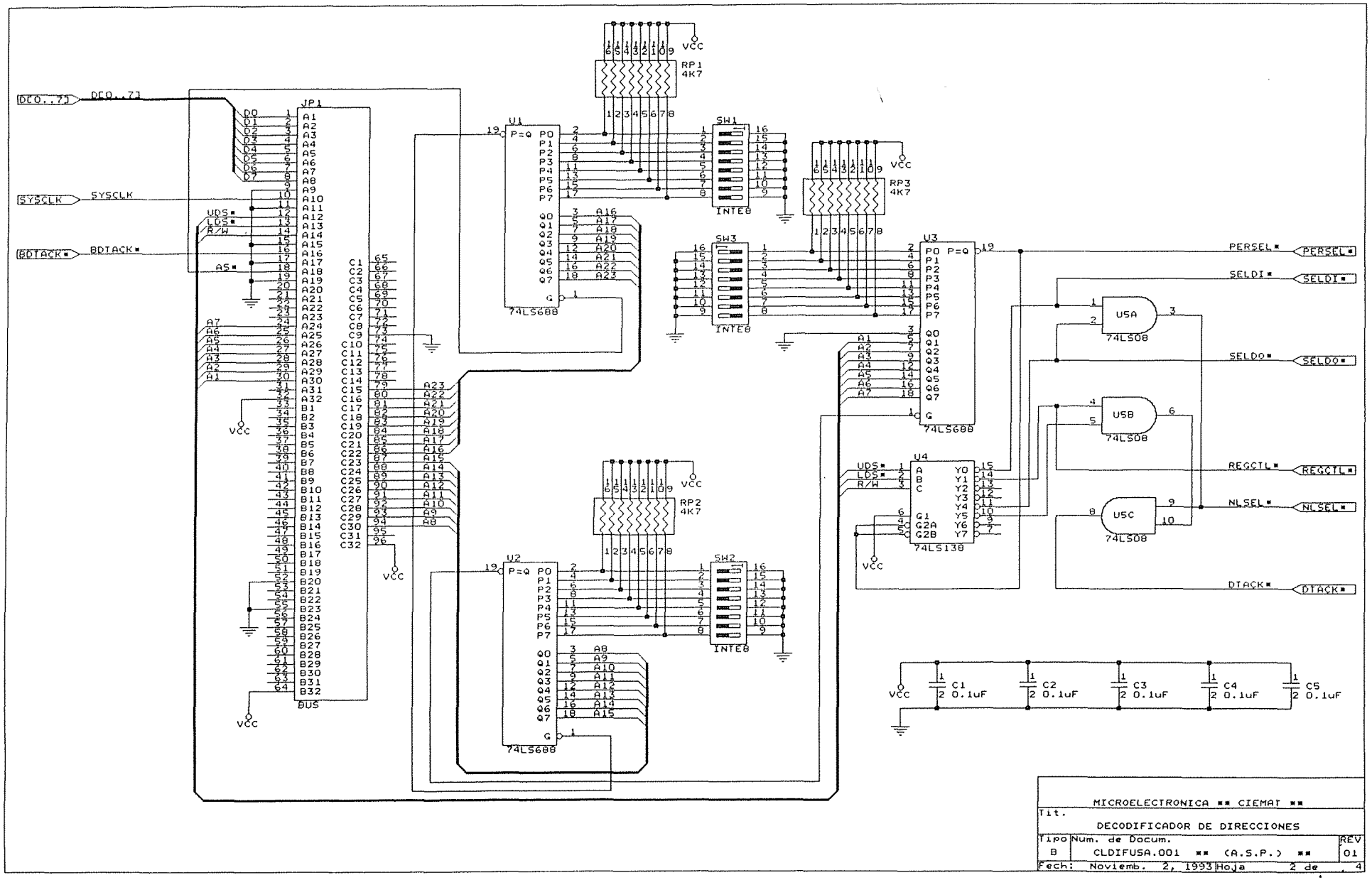

G11--G34=350Ω
CI 3A
OP470
G32
G33
500K
100K
G31
G34
R1
220Ω
-8V.
CI-3C
R18
10K
C6
0.1uF
C9
0.1uF
R21
10K
R15
82K
R12
82K
R9
82K- AE
C12
0.1uF
C3
0.1uF
+8V.
CI-3B
R6
82Ω
(19)
CI-2A
OP470
G22
G23
500K
100K
G21
G24
R2
220Ω
-8V.
CI-2C
R17
10K
AE
C5
0.1uF
C8
0.1uF
R20
10K- AE
R14
82K
R11
82K
R8
82K- AE
C11
0.1uF
C2
0.1uF
CI-2B
R5
82Ω
(21)
CI-1A
OP470
G12
G13
500K
100K
G11
G14
R3
220Ω
15
-8V.
CI-1C
R16
10K
AE.
C4
0.1uF
C7
0.1uF
R19
10K- AE
R13
82K
R10
82K
R7
10K- AE
C1
0.1uF
CI-1B
R4
82Ω- AE
(23)
R22
80Ω
1W
CI-1D
CI 4
REF. 02
C10
0.1uF
+6V.
Con. cable plano
40 pines.
Limite tarjeta.

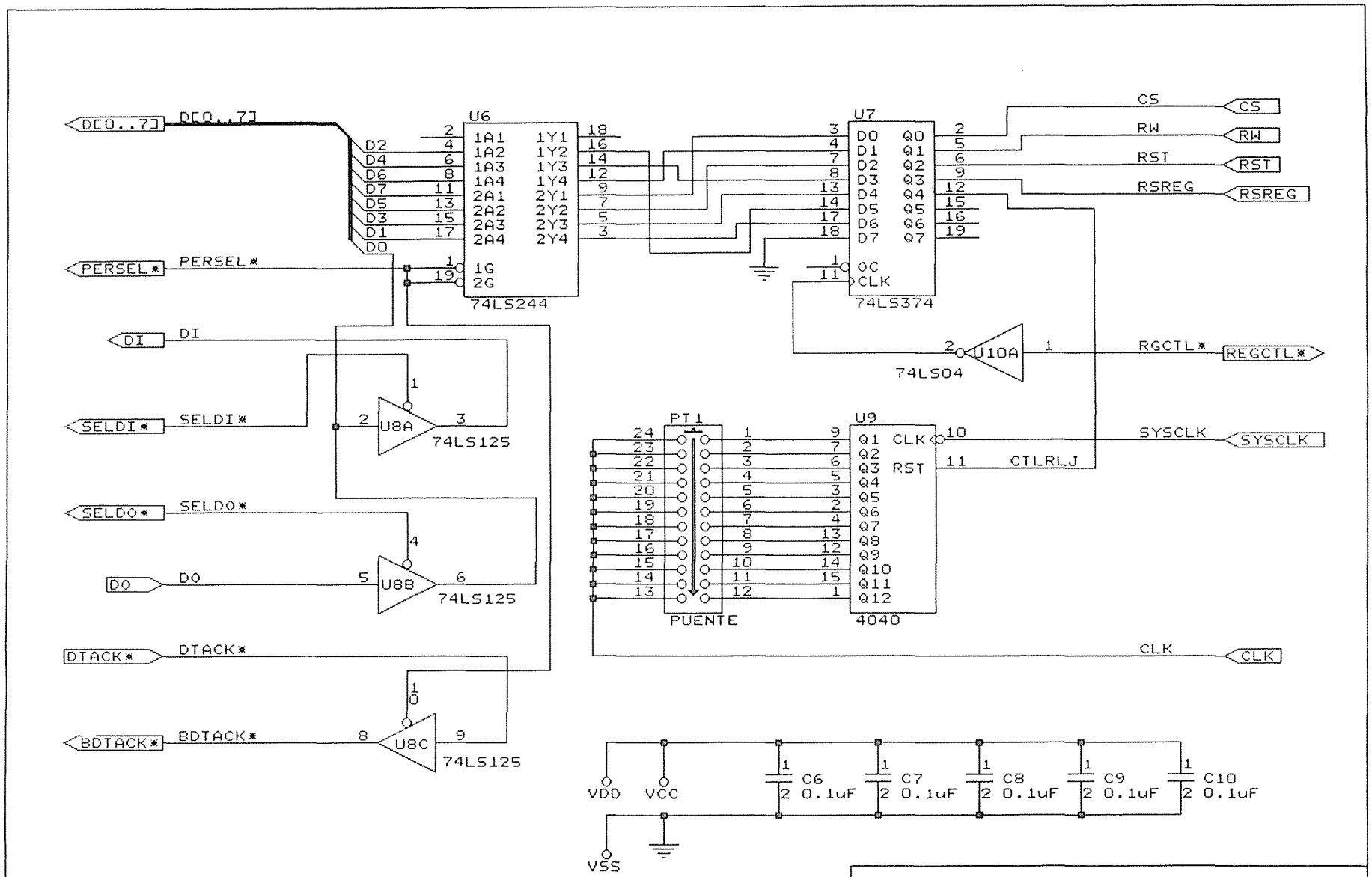

JP1
HEADER 14
14HV100
JP2
HEADER 14
14HV100
U1A
OP470
14DIP300
+8V
-8V
JP3
HEADER 26
26HH100
R22
33
RC20
U4
NC
VIN
TEMP
C10
0.1nF
CK05
+8V
+15V
-15V
U1B
U2B
U3B
C1
C2
C3
0.1uF
CK05
R10
R11
R12
82K
RC05
R7
R8
R9
R4
R5
R6
82
RC05
TSTPT2
UL1
SONIDO1
TSTPT2
UL3
SONIDO3
RC05
470K
R52
R62
47K
R53
R50
1K8
R60
C53
C61
100K
CK06
C54
C62
0.47K
CK06
Q2
Q5
BSX52A
T05
R51
R64
2K7
L4
L6
7uH
COIL200
C57
C64
2K2
R54
R65
100K
R56
R67
27K
R55
R68
1M
C56
C65
0.39K
C55
C66
Q3
Q6
L5
R58
R69
C34
C35
C36
C37
C38
C39
C40
C41
C42
C43
C44
C45
NOTAS:
OVD = Masa Digital
VCC = +5v
ANTONIO SERENA PEREZ
CIRCUITO GARRA
de Docum.
GARRA.001
REV
Jul. 11, 1991
Hoja
de


C2
0.7pF
R2
7M
U4
7805
VI
GND
VO
C13
0.1uF
C14
1uF
R1
2M8
C1
2.2nF
U1
356
R8
13K
R9
26K
R3
2M
C3
0.1uF
C4
1uF
C6
1uF
C5
0.1uF
J1
E1
U2
356
J3
S1
R10
26K
C7
0.1uF
C8
1uF
C10
1uF
C9
0.1uF
R11
13K
R4
13K
R6
78K
-15V
+15V
-15V
C11
0.1uF
C12
1uF
J2
S2
U3
357
R5
13K
R7
13k
MICROELECTRONICA ** Ciemat **
Tit.
ENTRADAS PARA CONTROLADOR BORROSO
Tipo
A
Num. de Docum.
BORROSO.SCH ** A.S.P. **
REV
00
Fech:
May 30, 1994
Hoja
1 de
1

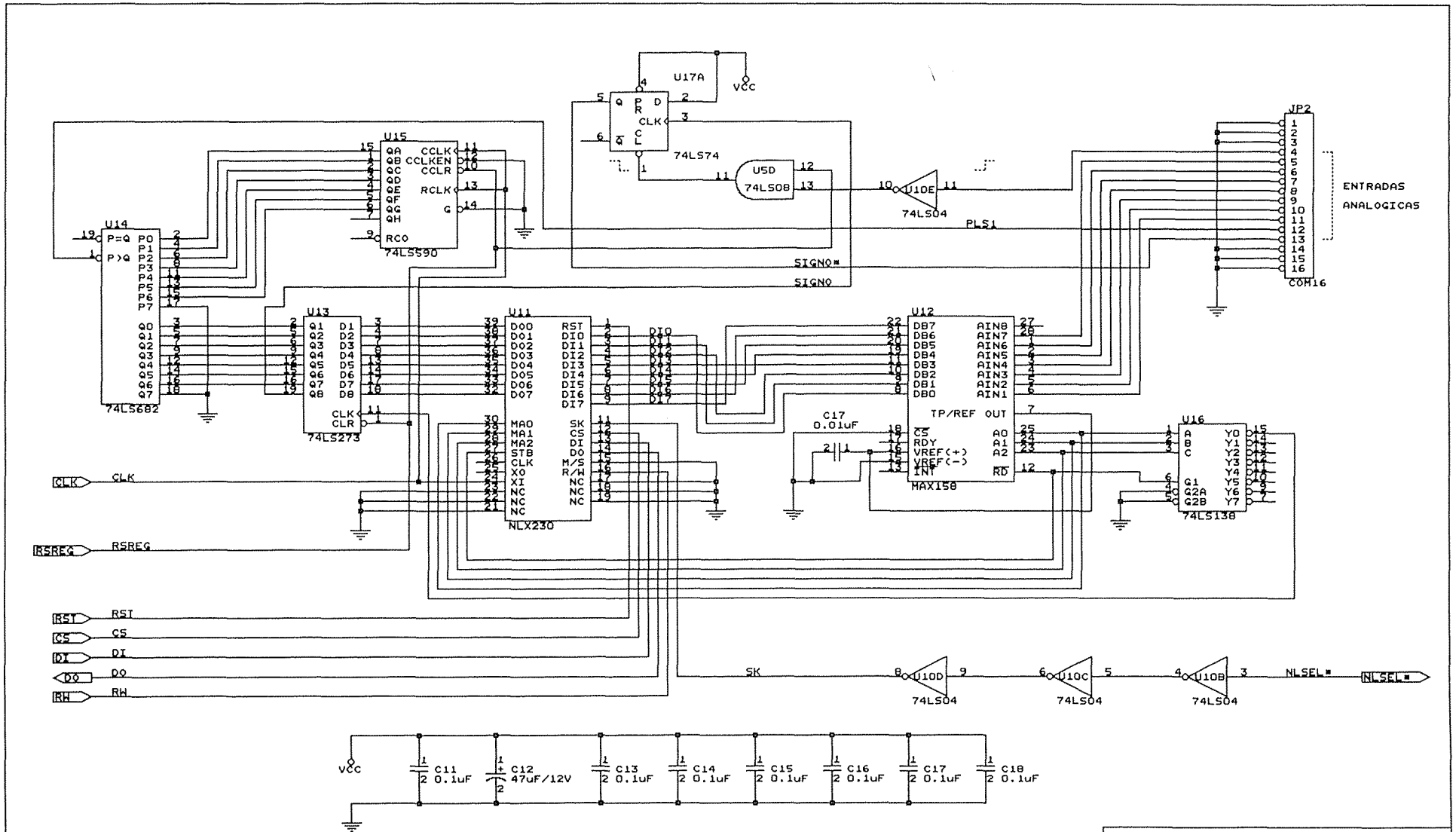


MICROELECTRONICA ** CIEMAT **		
Tit. CONTROLADOR LOGICA DIFUSA		
Tipo A	Num. de Docum. CLDIFUSA.ORG ** (A.S.P.) **	REV D1
Fech: Noviemb. 2, 1993	Hoja 1 de 4	


DC0..7J
SYSCLK
BDIACK*
AS*
UDS*
LDS*
R/W
JP1
BUS
U1
U2
U3
74LS688
RP1 4K7
RP2 4K7
RP3 4K7
SW1
SW2
SW3
INTE8
VCC
U4
74LS138
U5A
U5B
U5C
74LS08
PERSEL*
SELDI*
SELDO*
REGCTL*
NLSEL*
DTACK*
C1 0.1uF
C2 0.1uF
C3 0.1uF
C4 0.1uF
C5 0.1uF
MICROELECTRONICA ** CIEMAT **
DECODIFICADOR DE DIRECCIONES
CLDIFUSA.001 ** (A.S.P.) **
Noviemb. 2, 1993
2 de 4



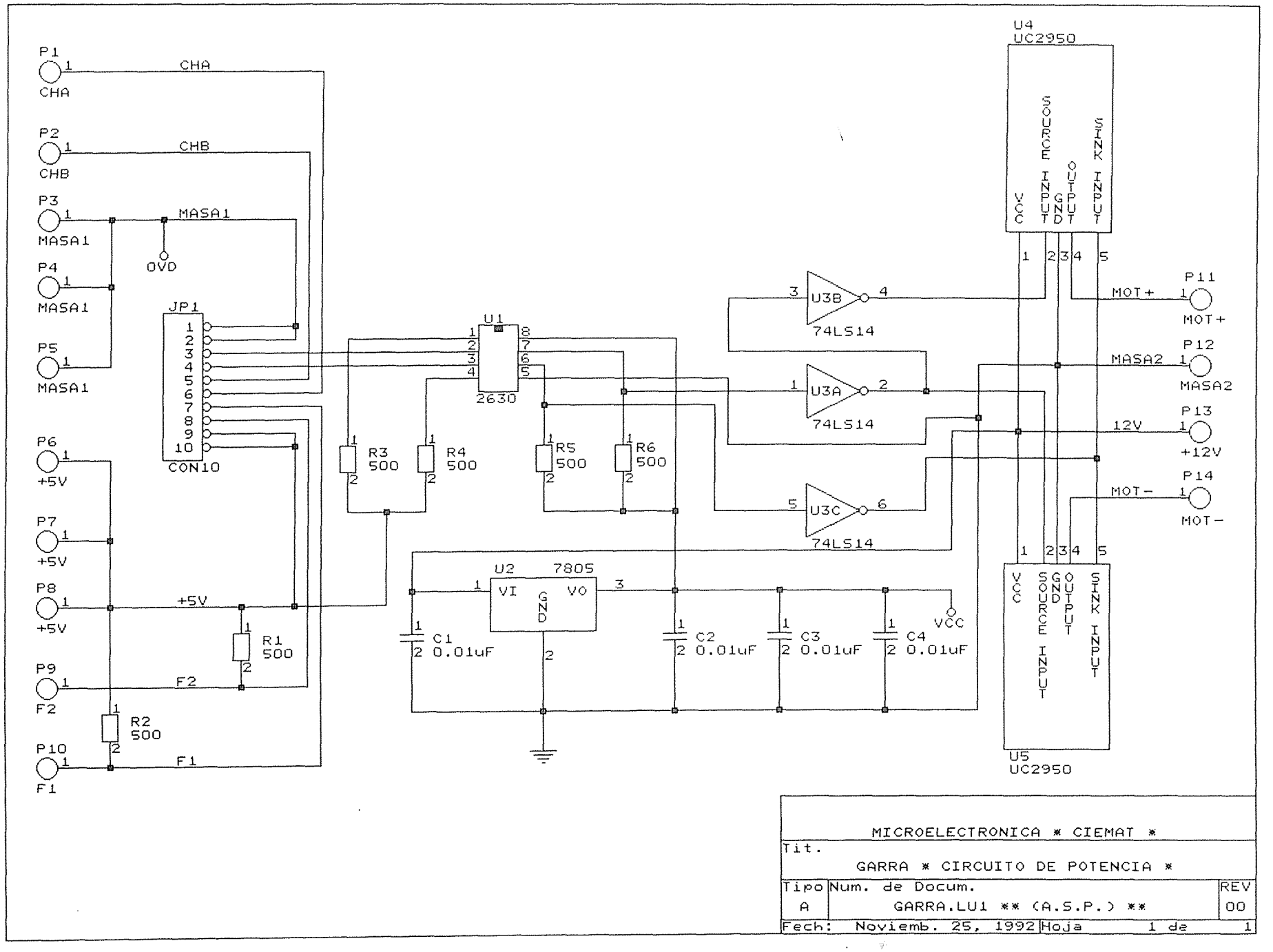
MICROELECTRONICA ** CIEMAT **		
Tit. ENTRADA DE DATOS		
Tipo	Num. de Docum.	REV
A	CLDIFUSA.002 ** (A.S.P.) **	01
Fech: Noviemb. 2, 1993	Hoja 3 de 4	



NOTAS:

C18 y C12 colocar en pata 26 de U12

MICROELECTRONICA ** CIEMAT **		
Tit. Controlador Logica Difusa		
Tipo B	Num. de Docum. CLDIFUSA.003 ** (A.S.P.) **	REV 02
Fech: May 24, 1994	Hoja 4 de 4	



MICROELECTRONICA * CIEMAT *		
Tit. GARRA * CIRCUITO DE POTENCIA *		
Tipo A	Num. de Docum. GARRA.LU1 ** (A.S.P.) **	REV 00
Fech: Noviemb. 25, 1992	Hoja 1 de 1	

Apéndice C. Ficheros de datos para programar el NLX230.

Datos para programar el microcontrolador NLX230.

```
 0 / 00 ;   1 / 00 ;   2 / 00 ;   3 / 00 ;   4 / 00 ;   5 / 00;
 6 / 00 ;   7 / 00 ;   8 / 00 ;   9 / 00 ;   a / 00 ;   b / 00;
 c / fe ;   d / ff ;   e / 00 ;   f / 00 ;  10 / 00 ;  11 / 00;
12 / 00 ;  13 / 00 ;  14 / 00 ;  15 / 00 ;  16 / 00 ;  17 / 00;
18 / 19 ;  19 / 00 ;  1a / 00 ;  1b / 00 ;  1c / 00 ;  1d / 00;
1e / 00 ;  1f / 00 ;  20 / 14 ;  21 / 1e ;  22 / 3c ;  23 / 1e;
24 / 46 ;  25 / 1e ;  26 / 5a ;  27 / 1e ;  28 / 6e ;  29 / 1e;
2a / 55 ;  2b / 3e ;  2c / 64 ;  2d / 3e ;  2e / 78 ;  2f / 3e;
30 / 8c ;  31 / 3e ;  32 / a0 ;  33 / 3e ;  34 / 00 ;  35 / 00;
36 / 00 ;  37 / 00 ;  38 / 00 ;  39 / 00 ;  3a / 00 ;  3b / 00;
3c / 00 ;  3d / 00 ;  3e / 00 ;  3f / 00 ;  40 / 00 ;  41 / 8b;
42 / 8b ;  43 / 28 ;  44 / 14 ;  45 / 05 ;  46 / 8b ;  47 / 8b;
48 / 28 ;  49 / 14 ;  4a / 05 ;  4b / 8b ;  4c / 86 ;  4d / 05;
4e / 05 ;  4f / 05 ;  50 / 8b ;  51 / 86 ;  52 / 00 ;  53 / 00;
54 / 00 ;  55 / 8b ;  56 / 86 ;  57 / 00 ;  58 / 00 ;  59 / 00;
5a / 00 ;  5b / 00 ;  5c / 00 ;  5d / 00 ;  5e / 00 ;  5f / 00;
60 / 00 ;  61 / 00 ;  62 / 00 ;  63 / 00 ;  64 / 00 ;  65 / 00;
66 / 00 ;  67 / 00 ;  68 / 00 ;  69 / 00 ;  6a / 00 ;  6b / 00;
6c / 00 ;  6d / 00 ;  6e / 00 ;  6f / 00 ;  70 / 00 ;  71 / 00;
72 / 00 ;  73 / 00 ;  74 / 00 ;  75 / 00 ;  76 / 00 ;  77 / 00;
78 / 00 ;  79 / 00 ;  7a / 00 ;  7b / 00 ;  7c / 00 ;  7d / 00;
7e / 00 ;  7f / 00 ;  80 / 00 ;  81 / 21 ;  82 / 41 ;  83 / 81;
84 / 01 ;  85 / 01 ;  86 / 22 ;  87 / 42 ;  88 / 82 ;  89 / 02;
8a / 02 ;  8b / 24 ;  8c / 44 ;  8d / 84 ;  8e / 04 ;  8f / 04;
90 / 28 ;  91 / 48 ;  92 / 88 ;  93 / 08 ;  94 / 08 ;  95 / 30;
96 / 50 ;  97 / 90 ;  98 / 10 ;  99 / 10 ;  9a / 00 ;  9b / 00;
9c / 00 ;  9d / 00 ;  9e / 00 ;  9f / 00 ;  a0 / 00 ;  a1 / 00;
a2 / 00 ;  a3 / 00 ;  a4 / 00 ;  a5 / 00 ;  a6 / 00 ;  a7 / 00;
a8 / 00 ;  a9 / 00 ;  aa / 00 ;  ab / 00 ;  ac / 00 ;  ad / 00;
ae / 00 ;  af / 00 ;  b0 / 00 ;  b1 / 00 ;  b2 / 00 ;  b3 / 00;
b4 / 00 ;  b5 / 00 ;  b6 / 00 ;  b7 / 00 ;  b8 / 00 ;  b9 / 00;
ba / 00 ;  bb / 00 ;  bc / 00 ;  bd / 00 ;  be / 00 ;  bf / 00;
c0 / 00 ;  c1 / 00 ;  c2 / 00 ;  c3 / 00 ;  c4 / 01 ;  c5 / 02;
c6 / 00 ;  c7 / 00 ;  c8 / 00 ;  c9 / 01 ;  ca / 02 ;  cb / 00;
cc / 00 ;  cd / 00 ;  ce / 01 ;  cf / 02 ;  d0 / 00 ;  d1 / 00;
d2 / 00 ;  d3 / 01 ;  d4 / 02 ;  d5 / 00 ;  d6 / 00 ;  d7 / 00;
d8 / 01 ;  d9 / 02 ;  da / 00 ;  db / 00 ;  dc / 00 ;  dd / 00;
de / 00 ;  df / 00 ;  e0 / 00 ;  e1 / 00 ;  e2 / 00 ;  e3 / 00;
e4 / 00 ;  e5 / 00 ;  e6 / 00 ;  e7 / 00 ;  e8 / 00 ;  e9 / 00;
ea / 00 ;  eb / 00 ;  ec / 00 ;  ed / 00 ;  ee / 00 ;  ef / 00;
f0 / 00 ;  f1 / 00 ;  f2 / 00 ;  f3 / 00 ;  f4 / 00 ;  f5 / 00;
f6 / 00 ;  f7 / 00 ;  f8 / 00 ;  f9 / 00 ;  fa / 00 ;  fb / 00;
fc / 00 ;  fd / 00 ;  fe / 00 ;  ff / 00 ;
```

```
/*
     Fichero : GARRABORROSA.C
     Fecha   : 4 - Febrero - 1994
     Descripcion : Este programa carga en el controlador nlx230
              el conjunto de reglas contenidas en un fichero.
*/

#include <stdio.h>

#include "nlx230.c"

#define   NUM_BYTES      256

main()
{
FILE *fp;
int i, numero, basura;
UCAR resultado, buffer[NUM_BYTES];
char fichero[20], carac1, carac2;

printf(" Cual es el fichero de datos ? ");
scanf( "%s" , fichero );
if ( (fp = fopen( fichero , "r" )) != NULL )
   {
   for ( i=0 ; i<NUM_BYTES ; i++ )
        {
        fscanf(fp,"%x %c %x %c",&basura,&carac1,&numero,
                                                &carac2);
        buffer[i] = (UCAR) numero;
        }
   if ( (resultado = cargar_nlx230( buffer )) != TRUE )
      {
     printf("Error 1: Los datos cargados no se corresponden");
       printf(" con los leidos");
      }
   else
      {
      printf(" Mensaje 1: Datos cargados correctamente");
      CONTROL = 0x48;
      CONTROL = 0x68;
      CONTROL = 0x60;
      CONTROL = 0x70;
      }
   fclose( fp );
   }
else
   printf(" Error 2: No encuentro ese fichero");

}
```

```
/*
     Fichero : NLX230.C
     Fecha :   4 - Febrero - 1994
     Descripcion : libreria de funciones para manejar el
                   controlador borroso nlx230 de la tarjeta de
                   VME.
*/

#define   CONTROL    (*(char *) 0x100001) /* Registro de control*/
                          /* CS R/W RST RSREG CTRLLJ X X X */
#define   REG_DATOS (*(short  *)  0x100000)  /*Direccion  del
                                                 nlx230*/

#define   TRUE       1
#define   FALSE      0

typedef   unsigned char  UCAR;
typedef   char           CAR;

/*
     Esta funcion lee la informacion contenida en el nlx230, la
     coloca en una tira tipo caracter y devuelve un puntero a esa
     tira.
*/

UCAR *leer_nlx230()
{
UCAR mascara, buffer[256];
short int entero, i, dato;

CONTROL = 0x08;
CONTROL = 0x28;
CONTROL = 0xE8;
for(i=0;i<256;i++)
     {
     buffer[i] = 0;
     for(mascara=0x80;mascara>0;mascara>>=1)
          {
          dato = 0x01 & REG_DATOS;
          buffer[i] |= (UCAR) (dato ? mascara : 0);
          }
     }

CONTROL = 0x68;
for(i=0;i<2;i++)
     entero = REG_DATOS;

return( buffer );
}
```

```
/*
     Esta funcion contrasta los datos contenidos en una tira tipo
     caracter con los que tiene el nlx230.
*/


char comprobar_nlx230( UCAR *buffer_original )
{
UCAR *buffer_chip;
int i;

buffer_chip = leer_nlx230();

for(i=0;i<256;i++)
     if ( *(buffer_chip + i) != *(buffer_original + i) )
          return( FALSE );

return( TRUE );
}


/*
     Esta funcion carga el nlx230 con la informacion suministrada
     en una tira tipo caracter.
*/

char cargar_nlx230( UCAR *buffer )
{
int i;
UCAR mascara;

CONTROL = 0x08;
CONTROL = 0x28;
REG_DATOS = 0;
CONTROL = 0xA8;
REG_DATOS = 0;

for(i=0;i<256;i++)
     for(mascara = 0x80;mascara > 0; mascara >>= 1)
       REG_DATOS = ( ( (*(buffer + i)) & mascara ) ? 0x01 : 0 );

CONTROL = 0x28;
CONTROL = 0;
CONTROL = 0;

return( comprobar_nlx230( buffer ) );
}
```