UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

DIVISIÓN DE ESTUDIOS DE POSGRADO DE LA FACULTAD DE INGENIERÍA

DISEÑO Y CONSTRUCCIÓN DE UNA NEUROCOMPUTADORA ANALÓGICA

T E S I S

QUE PARA OBTENER EL GRADO DE:
MAESTRO EN INGENIERÍA *electrica*
P R E S E N T A

MIGUEL ANGEL BAÑUELOS SAUCEDO

DIRECTOR DE TESIS: M. en C. JOSÉ LUIS PÉREZ SILVA

LABORATORIO DE ELECTRÓNICA
CENTRO DE INSTRUMENTOS

MÉXICO, D.F. 1997

A mis padres: L. Angel Bañuelos Gutiérrez
y María de Lourdes Saucedo de Bañuelos,
quienes son para mí el mejor ejemplo
de amor y dedicación.

A mis hermanos: Angel Leonardo Bañuelos Saucedo
y Lourdes Angélica Bañuelos Saucedo,
mis grandes amigos
de toda la vida.

AGRADECIMIENTOS

A mi maestro José Luis Pérez Silva, por haberme invitado al Centro de Instrumentos, donde se desarrolló este trabajo, pero en especial por su invaluable amistad y por el ejemplo que me da tanto profesionalmente como por su calidad humana.

A Wilfredo Martínez y José Castillo, con quienes estoy orgulloso de colaborar, de tener su amistad y de quienes constantemente sigo aprendiendo.

A mis amigos y compañeros del Centro de Instrumentos: Alejandro Padrón, Sergio Quintana; a Alberto Herrera por la minuciosa revisión de este trabajo; a Antonio Garcés, Andrea Miranda, Rosendo Fuentes y Gerardo Calva. A Etna Cervantes, por sus comentarios al manuscrito y asesoría en la programación. A Claudia Licea por su colaboración en el montaje de los circuitos impresos.

A Francisco Rodríguez, por su continua amistad y apoyo desde mis primeros pasos por la Licenciatura.

A los miembros del jurado: Dr. Felipe Lara, M. en C. José Luis Pérez, M. en C. Luis Marcial Hernández, M. en C. Alberto Herrera y M. en I. Jorge Rodríguez, por sus valiosos comentarios.

Al Dr. Claudio Firmani, el Dr. Felipe Lara y la Dra. Nydia Lara por su apoyo al desarrollo de la investigación en redes neuronales artificiales.

A la Dirección General de Asuntos del Personal Académico por el apoyo económico para la realización de este proyecto.

Al Consejo Nacional de Ciencia y Tecnología por la beca otorgada para realizar mis estudios de Maestría.

INDICE

INTRODUCCION

INTRODUCCION

El campo de investigación en redes neuronales artificiales surge como una búsqueda por desarrollar sistemas autónomos capaces de operar independientemente del control humano y dentro de ambientes inciertos o desestructurados. Aunque se ha inspirado en los sistemas biológicos, el objetivo es crear morfologías neuronales (sistemas de procesamiento paralelo) que aun sin corresponder a las observadas en la biología, sean capaces de implementar diversas funciones computacionales.

Las áreas de investigación dentro del campo de las redes neuronales artificiales son muy diversas e incluyen: la elaboración de nuevos modelos de neuronas, el diseño de nuevas topologías, el análisis matemático de sistemas neuronales, el desarrollo de aplicaciones, la creación de métodos de ajuste o entrenamiento y las implantaciones (electrónicas, ópticas, etc.).

Las redes neuronales artificiales son sistemas de procesamiento paralelo que han demostrado ser útiles en tareas de clasificación de patrones, regeneración de patrones, optimización y control. Por lo tanto, cada vez es mayor el interés por desarrollar implantaciones electrónicas, las cuales puedan explotar efectivamente las capacidades del procesamiento paralelo. Estas implantaciones se inspiran en modelos simplificados de neuronas, que sin embargo representa una propuesta válida, dado que se considera que el poder computacional de los sistemas neuronales estriba en la interconectividad y se asume que un nodo individual realiza operaciones sencillas.

Conscientes de la potencialidad de las redes neuronales, y en especial de las redes neuronales dinámicas, en el Centro de Instrumentos de la UNAM, se ha venido trabajando en el estudio de ellas, con el objetivo de generar nuevos modelos de neuronas y nuevas topologías de redes que se puedan aplicar en sistemas de control y/o procesamiento de señales. El análisis de estos nuevos modelos de neurona se puede apoyar fuertemente en simulaciones por computadora; sin embargo, dado que se trata de sistemas dinámicos no-lineales, aún las redes más simples requieren de un gran tiempo de cómputo. Es por ello que se ha recurrido al diseño de modelos electrónicos analógicos de neuronas, los cuales permiten obtener resultados en tiempo real. Estos últimos presentan sin embargo dificultades en el momento de ajustarlos, ya que normalmente se requiere desconectar cada componente en el momento de ajustarse, para así conocer el valor del parámetro correspondiente. Como alternativa se presenta el control digital de los parámetros del modelo pero manteniendo un procesamiento analógico, es decir, desarrollar un modelo híbrido.

OBJETIVO

Con la finalidad de facilitar el estudio de redes neuronales dinámicas pequeñas mediante la simulación en tiempo real, pero con la posibilidad de un control fácil y preciso de los parámetros, se propone diseñar y construir una neurocomputadora analógica para redes neuronales cuyos parámetros se controlen digitalmente desde una computadora personal.

La neurocomputadora deberá de ser capaz de implementar algunos de los modelos con procesamiento temporal de mayor interés, tales como el integrador con fugas y el integrador con disparo, así como la funciones de activación más usadas: escalón, rampa y sigmoidal. Adicionalmente deberá existir la posibilidad de generar respuestas de tipo pulso.

El uso de una computadora personal para controlar los parámetros de la computadora analógica deberá permitir un fácil ajuste de los mismos, así como la generación de archivos de parámetros que permitan la rápida reprogramación de la computadora.

CAPITULO I

ANTECEDENTES

1.1 ANTECEDENTES

El cerebro humano contiene un gran número de procesadores elementales llamados neuronas, se cree que contiene 10^{11}, cada una de las cuales está conectada con hasta otras 10, 000 [1]. Es decir, constituye una increíble red de procesamiento, donde la información se transmite mediante impulsos eléctricos. Los primeros intentos por entender el funcionamiento del cerebro, dieron como resultado algunos modelos elementales del comportamiento de las neuronas.

En la figura 1 se muestra un esquema simplificado de una neurona, en donde se indican las principales partes de la misma. Las dendritas se encargan de recibir los impulsos eléctricos de otras neuronas en puntos de contacto denominados *sinápsis*. Como resultado de la llegada de estos impulsos, la neurona puede generar un impulso de salida (disparo), el cual envía por el axón hacia otras neuronas. De esta manera, podemos ver a las dendritas como las terminales de entrada de la neurona y al axón como la terminal de salida. Las sinápsis se forman generalmente entre el axón de una neurona y la dendrita de otra, aunque se pueden encontrar otras combinaciones [1]. Algunas de las sinápsis son excitatorias, es decir tienden a promover el disparo de la neurona, y otras inhibitorias, es decir tratan de impedir el disparo [1].

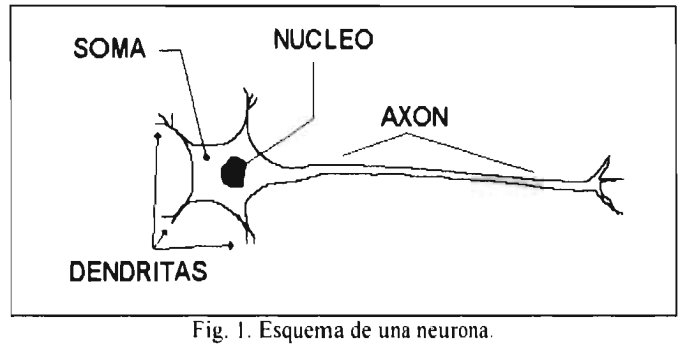
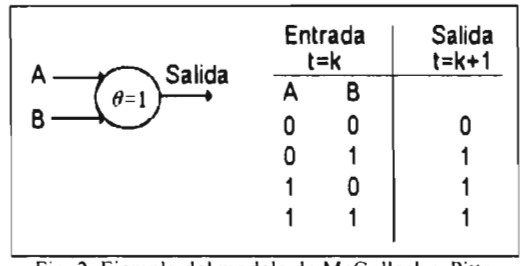


Fig. 1. Esquema de una neurona.

En 1943, McCulloch y Pitts desarrollaron un modelo matemático el cual se basaba en el carácter "todo o nada" (es decir, disparo o no disparo), de la actividad nerviosa [2]. En su modelo consideran tanto sinápsis excitatorias como inhibitorias. Para que la neurona dispare, debe existir un cierto número de entradas excitatorias activas, mientras que la presencia de una sola entrada inhibitoria evita que la neurona dispare. Esta idea la podemos observar en la figura 2.



Entrada t=k		Salida t=k+1
A	B	
0	0	0
0	1	1
1	0	1
1	1	1

Fig. 2. Ejemplo del modelo de McCulloch y Pitts.

En la neurona de la figura 2, se tienen dos entradas excitatorias A y B. Para que la salida se dispare (salida=1), al menos debe existir una entrada excitatoria, lo cual se denota como $\theta=1$ (a θ se le denomina umbral [3]). La salida ocurre una unidad de tiempo después de leerse las entradas y de esta manera hay una ligera correspondencia con el retardo observado en las neuronas biológicas [2].

El modelo propuesto por McCulloch y Pitts se ha ampliado y en la literatura reciente se puede encontrar definido como sigue[1] [4],[5]:

$$y = \psi\left(\sum_{i=1}^{n} w_i x_i + \Theta\right) \qquad (1),$$

donde **y** es la salida de la neurona con **n** entradas, $\Psi(\bullet)$ es una función no-lineal comúnmente llamada *función de activación*, $\mathbf{x}_i$ es el valor de la i-ésima entrada a la neurona, $\mathbf{w}_i$ es la ganancia o peso de cada entrada y θ el valor de umbral.

Dicho de otra manera, una neurona puede ser vista como un elemento de procesamiento de información, el cual recibe un vector de entrada *n*-dimensional

$$\mathbf{x}(t) = \left[x_1(t), x_2(t), \ldots, x_i(t), \ldots, x_n(t)\right]^T \in \Re^n,$$

y produce una salida escalar

$$y(t) \in \Re^1.$$

la capacidad de procesamiento de la neurona esta dada por el mapeo no-lineal Ne tal que

$$y(t) = Ne\left[\mathbf{x}(t) \in \Re^n\right] \in \Re^1.$$

El mapeo no-lineal puede ser dividido en dos partes: confluencia y activación no-lineal. La operación de confluencia proporciona las operaciones de ponderado, agregado y umbral. La operación de umbral la podemos incluir si definimos los vectores de entrada y pesos ampliados como sigue:

[1] Cabe aclarar que actualmente no existe uniformidad en lo que se denomina modelo de McCulloch y Pitts.

$$\mathbf{x}_a(t) = \left[x_0(t), x_1(t), \ldots, x_i(t), \ldots, x_n(t)\right]^t \in \Re^{n+1},$$

donde $x_0(t) = 1$, y

$$\mathbf{w}_a(t) = \left[w_0(t), w_1(t), \ldots, w_i(t), \ldots, w_n(t)\right]^t \in \Re^{n+1}.$$

donde $w_0(t)$ introduce un término de umbral (bias) en la operación de confluencia.

La operación de confluencia representa el ponderado de las señales de entrada, $\mathbf{x}_a(t) \in \Re^{n+1}$, con el conocimiento almacenado en las sinápsis, $\mathbf{w}_a(t)$, y la suma espacio-temporal de esas entradas ponderadas. El operador de confluencia puede verse como un mapeo lineal del espacio (n+1)-dimensional $\mathbf{x}_a(t) \in \Re^{n+1}$ al espacio unidimensional $u(t) \in \Re^1$ tal que

$$u(t) = \mathbf{w}_a(t) \copyright \mathbf{x}_a(t),$$

donde © es el operador de confluencia.

Por otro lado, la activación no-lineal mapea el valor de confluencia $u(t) = \in (-\infty, \infty)$ a una salida acotada, esto es,

$$y(t) = \Psi\left[u(t)\right],$$

o bien,

$$y(t) = \Psi\left[\mathbf{w}_a \copyright \mathbf{x}(t)\right] \in \Re^1.$$

donde $\Psi[\bullet]$ es la función de activación no-lineal y(t) es la salida, que en general está normalizada a un valor en el intervalo [0,1] o bien [-1,1]. En la figura 3 se muestra una representación del modelo generalizado de neurona [6].

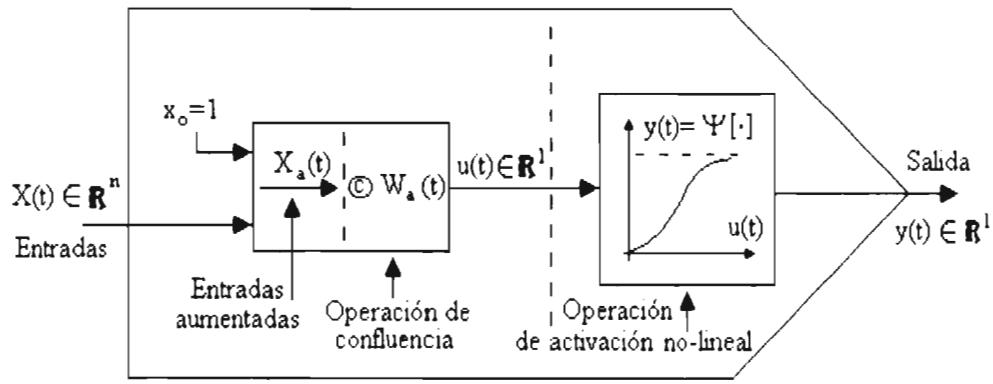


Fig. 3. Modelo de neurona generalizado.

Si la función no-lineal Ψ[•] la definimos como un limitador duro (del inglés hard limiter), cuya gráfica entrada-salida se muestra en la figura 4, la salida sólo tomará alguno de dos valores, en similitud a la propuesta original de McCulloch y Pitts. Sin embargo como (1) representa un modelo general, la funcion no-lineal Ψ[•] puede tomar multitud de variantes [5], siendo algunas de las más conocidas el limitador simple y la tangente hiperbólica (ver fig. 5).

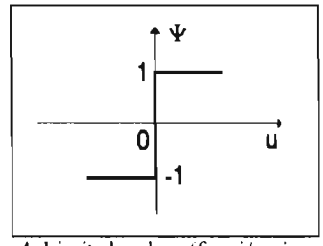


Fig. 4. Limitador duro (función signum).

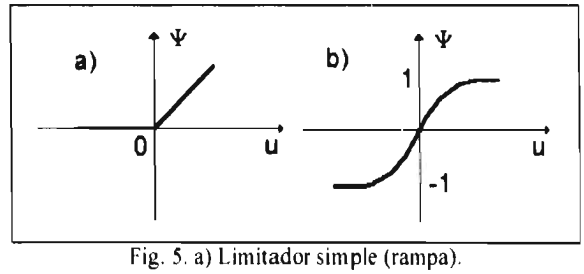


Fig. 5. a) Limitador simple (rampa).
b) Tangente hiperbólica (sigmoidal).

Es en el marco de este modelo de neurona generalizado que se ubica el desarrollo de una neurocomputadora analógica, donde como se mencionó en la introducción se diseñará un modelo electrónico de neurona con parámetros controlados digitalmente.

I.2 BIBLIOGRAFIA

[1] C. F. Stevens. "The neuron". *Scientific American.* 1979.

[2] W. S. McCulloch, W. Pitts. "A logical calculus of the ideas immanent in nervous activity". *Bulletin of Mathematical Biophysics.* 5: 115-133. 1943.

[3] M. L. Minsky. *Computation: finite and infinite machines.* Prentice-Hall, Inc. 1967.

[4] J. Hertz, A. Krogh, R. G. Palmer. *Introduction to the theory of neural computation.* Addison-Wesley Publishing Company. 1991.

[5] A. Cichocki, R. Unbehauen. *Neural networks for optimization and signal processing.* John Wiley and Sons. 1993.

[6] M. Gupta & D. Rao. "Neuro-control systems: a tutorial". in *Neuro-control systems: Theory and applications.* M. Gupta & D. Rao editors. IEEE Press. pp. 1-43. 1994.

II DESARROLLO ELECTRONICO

II . 1 INTRODUCCION

Con el propósito de facilitar el estudio de redes neuronales pequeñas, se ha propuesto el desarrollo de una computadora analógica con parámetros controlados digitalmente. El objetivo es poder utilizar la rapidez de procesamiento inherente a los circuitos analógicos, pero conservando un control digital sobre los parámetros más importantes de la red, con el fin de facilitar su ajuste y configuración.

II . 2 ARQUITECTURA DEL SISTEMA

La computadora se diseñó en módulos funcionales (neuronas), los cuales se pueden interconectar para generar la topología de red deseada. Los módulos a su vez están subdivididos en distintos bloques encargados de implementar los diferentes procesos del modelo de neurona. Aquí cabe mencionar que el modelo de neurona no intenta representar fielmente a una neurona biológica y se basa principalmente en el trabajo desarrollado por los grupos de Electrónica y Neurocontrol del Centro de Instrumentos.

En la figura 1 se muestra un esquema de la computadora analógica para redes neuronales donde se observan sus dos componentes fundamentales: la computadora analógica y la computadora personal. La computadora analógica consiste de 10 módulos (neuronas) y una interfaz de control y comunicación con la computadora personal. Cada uno de los módulos está formado a su vez de 4 bloques que corresponden a cada una de las etapas del modelo de neurona propuesto. Los bloques cuentan con bornes mediante los cuales se puede alambrar la topología de red deseada, de acuerdo con las posibilidades que ofrece el sistema.

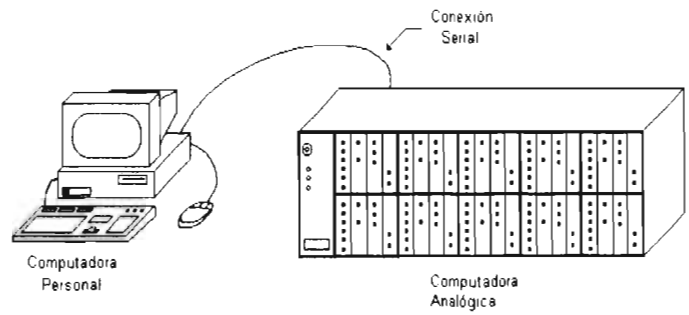


Fig. 1. Computadora analógica para redes neuronales.

La computadora personal se encarga de servir de interfaz entre el usuario y la computadora analógica en las tareas de especificación de los parámetros de los módulos. Para desarrollar

el programa de control, se eligió Visual Basic por la facilidad que proporciona para desarrollar aplicaciones gráficas bajo ambiente Windows, en comparación con otras herramientas de que se disponía como Visual C++ y Borland C++. Para la comunicación entre la computadora analógica y la computadora personal se eligió la de tipo serial debido a que en las computadoras personales normalmente se encuentra disponible un puerto serial y para la aplicación no se requería una alta velocidad de transmisión de la información. Además, dado que se eligió a Visual Basic como lenguaje de programación, el manejo del puerto serial es más sencillo que el de algún puerto paralelo, debido principalmente a los candados que impone Windows para el manejo de los recursos del sistema.

II . 3 MODELO DE NEURONA

El modelo de la neurona se ha definido de acuerdo a los modelos más usados en la literatura y se encuentra subdividido en cuatro etapas: Sumador, Integrador, Función de Activación y Disparo, según se muestra en la fig. 2. Cada una de estas etapas estará integrada en una tarjeta que se conectará a la computadora analógica mediante un slot de tipo peine. De esta manera se facilitará el mantenimiento, prueba y modificación del diseño.

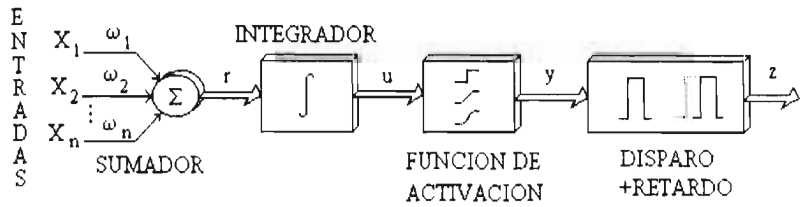


Fig. 2. Diagrama de bloques del modelo de neurona.

La etapa sumadora se encarga de generar la suma de las entradas ponderadas, es decir multiplicadas por un peso, logrando con ello implantar una sinápsis que realice una suma espacial. De esta manera se trata de hacer un símil con la suma espacial que se realiza en las sinápsis de la neurona [1].

El Integrador proporciona un procesamiento temporal a la señal, de tal manera que no sólo influya el valor de las entradas presentes, si no también se tome en cuenta la historia de las mismas. Esto permite a la neurona decodificar los trenes de pulsos provenientes de otras neuronas. Para este modelo de neurona se proponen el integrador con fugas y su variante denominada integrador con disparo. El integrador con fugas es un integrador que evita la saturación mediante un mecanismo continuo de descarga del potencial. El integrador con disparo (o integrador y disparo) es una variante en la que al cumplirse una condición de disparo de la actividad de la neurona, inmediatamente se lleva al integrador a condiciones iniciales. Esto último forza a que el siguiente disparo ocurra hasta que el potencial del integrador alcance nuevamente la condición de disparo, lo cual correspondería a lo que en una neurona biológica sería el período refractario.

La Función de Activación consiste en una función no-lineal. Para nuestro modelo, hemos seleccionado las funciones escalón, rampa saturación y tangente hiperbólica, por ser de las más encontradas en la literatura. Sin embargo, el diseño modular propuesto es suficientemente flexible para aceptar el reemplazo de bloques y, por ende, la incorporación de otras funciones de activación.

La etapa de Disparo consiste en un circuito generador de pulsos, como una forma de replicar los impulsos que generan las neuronas biológicas. Además el pulso se puede retardar para representar el retraso que sufre el pulso al viajar por el axón de la neurona.

En el diseño de cada una de las etapas, se debe tomar como objetivo el lograr que los parámetros del modelo sean controlables digitalmente, debido a que esto facilitará la repetibilidad de los experimentos. Esto se puede conseguir mediante convertidores digital-analógicos (DAC, por sus siglas en inglés), en especial algunos que trabajan bajo el esquema de escalera R-2R [2], ya que con ellos se puede implementar un divisor de voltaje controlado digitalmente. Se consideró apropiado utilizar convertidores de 8-bits porque proporcionan una resolución aceptable para los parámetros y para limitar el costo y complejidad del sistema.

En la fig. 3 se muestra el diagrama de la conexión escalera R-2R de un DAC de 8 bits. Del análisis de esta configuración se puede obtener que el voltaje a la salida está dado por

$$V_{OUT} = \left(\frac{D7}{2} + \frac{D6}{4} + \frac{D5}{8} + \frac{D4}{16} + \frac{D3}{32} + \frac{D2}{64} + \frac{D1}{128} + \frac{D0}{256} \right) V_{REF} \qquad (1),$$

donde D7 a D0 representan los bits de la palabra de control (Dx=0 ó Dx=1). En lugar de utilizar un voltaje de referencia constante, como sería el caso en una aplicación estándar del DAC, podemos aplicar una señal Vx variante en el tiempo y, entonces la red escalera R-2R se convertirá en un atenuador controlado digitalmente. Es esta propiedad la que explotamos para poder ajustar digitalmente los parámetros del modelo de neurona.

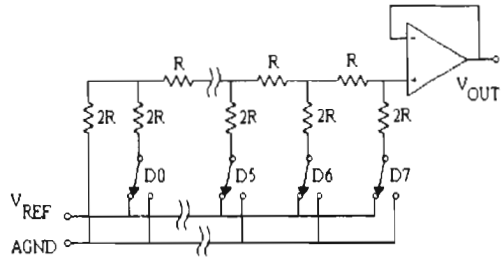
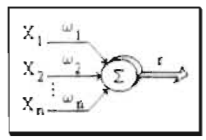


Fig. 3. Diagrama simplificado de un convertidor digital-analógico tipo escalera R-2R.

Con la finalidad de minimizar el tamaño de los circuitos impresos, se optó por utilizar el circuito integrado (C.I.) MAX505 del fabricante MAXIM. Este circuito es un DAC cuádruple con salida de voltaje (incluye un buffer seguidor para cada uno de los DACs) con excursión de riel a riel ("rail to rail swing"). Este circuito integrado tiene la ventaja de contar con referencias de voltaje independientes para cada DAC, lo cual lo posibilita para emplearse como cuatro divisores de voltaje controlados digitalmente. Se desecharon otras opciones como el AD7225 de Analog Devices (DAC cuádruple) debido a que las terminales de voltaje de referencia sólo aceptaban voltajes mayores a 2 V, lo cual impedía su empleo en esta aplicación debido a que se necesitaba aplicar en estas terminales potenciales que incluyeran al cero; o como el AD7228 de Analog Devices, el cual es un DAC óctuple pero con una referencia de voltaje común a todos los convertidores.

El seleccionar al circuito integrado (C.I.) MAX505 representa una excelente alternativa en cuanto a densidad de componentes; sin embargo, va a producir límites importantes para el diseño general de la computadora analógica. Los principales límites que quedan fijados por el MAX505 son los voltajes de alimentación y las limitaciones del ancho de banda (incluyendo slew-rate). El C.I. está diseñado para operar con ±5 V de alimentación, si bien puede operar con fuentes de ±6 V. Con la finalidad de trabajar con el mayor voltaje de alimentación posible para minimizar los efectos de saturación de los amplificadores operacionales, se decide diseñar la electrónica para operar con ±6 V. De esta manera, se trata de dar un margen que permita garantizar que se podrá procesar señales de hasta ±5 V. Existe la posibilidad de sustituir la totalidad de los amplificadores operacionales por versiones que pueden trabajar con excursión de fuente a fuente, sin embargo esto incrementaría el costo del sistema. No obstante, de esta manera sería posible incrementar el nivel de excursión de las señales y con ello mejorar la relación señal-ruido.

Otra limitación importante, como se mencionó, es el ancho de banda. El C.I. MAX505 tiene un producto ganancia-ancho de banda de 1 MHz y un slew-rate de 3 V/μs. Estas dos características limitan el ancho de banda total del sistema y condicionan las especificaciones que deberán tener el resto de los amplificadores operacionales utilizados. Es importante enfatizar que para cumplir los objetivos propuestos, no resulta indispensable que la computadora analógica sea capaz de procesar señales de frecuencias muy altas. Por ejemplo, los impulsos eléctricos que generan las neuronas biológicas pueden estar en un intervalo de frecuencias que va desde algunos Hertz a algunos cientos de Hertz [1]. Como se verá, el sistema implementado satisface esta última característica.



II.3.1 ETAPA SUMADORA

La primera etapa del modelo de neurona propuesto es la etapa sumadora. En esta etapa se realiza la suma ponderada de las entradas a la neurona. Cada una de las entradas es multiplicada por un factor de ganancia conocido comúnmente como peso. De forma general, el sumador realiza la siguiente operación

$$r = \sum_{i}^{n} \omega_i x_i \qquad (2),$$

donde ω_i son los pesos de las x_i entradas a la neurona. Los pesos pueden ser tanto de signo positivo como de signo negativo, con lo cual se pueden implementar entradas excitadoras o inhibitorias respectivamente. Por otro lado, es común que el valor de estos pesos esté normalizado a un intervalo entre -1.0 y 1.0.

Existen diferentes configuraciones típicas para la implantación de circuitos sumadores, tanto inversores como no-inversores. Para la construcción de esta etapa se utilizó un sumador inversor típico como el que se muestra en la fig. 4.

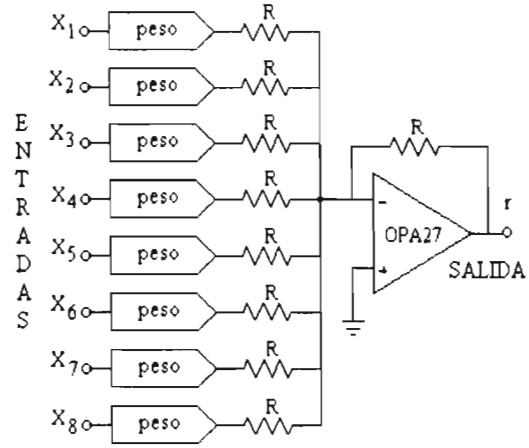


Fig. 4. Esquema del circuito sumador

Cada una de las entradas al sumador está precedida por una etapa de ajuste de ganancia, la cual está implementada con la configuración mostrada en la fig. 5.

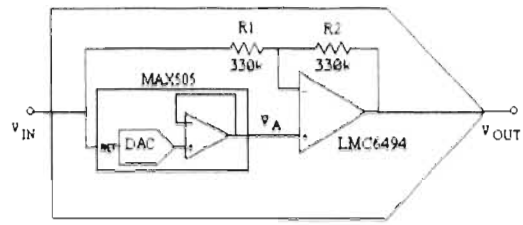


Fig. 5. Generación de pesos positivos y negativos.

La configuración diseñada resulta interesante porque, mediante el uso de un amplificador operacional externo, los DACs pueden generar pesos tanto positivos como negativos. Para este circuito (ver figura 5), se puede calcular que la salida del amplificador operacional está dada por

$$V_{OUT} = \left(\frac{R_1 + R_2}{R_1}\right)V_A - \left(\frac{R_2}{R_1}\right)V_{IN} \qquad (3).$$

como las resistencias R_1 y R_2 son iguales se tiene

$$V_{OUT} = 2V_A - V_{IN} \qquad (4),$$

donde

$$V_A = \frac{N_D}{256}V_{IN} \qquad (5),$$

donde N_D es el código digital de 8-bits que recibe el DAC.

Sustituyendo (5) en (4) se obtiene

$$V_{OUT} = V_{IN}\left[\frac{N_D}{128} - 1\right] \qquad (6).$$

donde si $N_D = 255$ se tendría $V_{OUT} = 0.992V_{IN}$,

si $N_D = 128$ se tendría $V_{OUT} = 0$,

si $N_D = 0$ se tendría $V_{OUT} = -V_{IN}$;

por lo cual, dependiendo del código, el voltaje de salida estará dentro del intervalo

$$-V_{IN} \leq V_{OUT} \leq 0.992 V_{IN} \quad (7),$$

y, por lo tanto, los pesos se podrán variar entre -0.992 y 1.0, debido a que se utiliza un sumador inversor. Como se puede ver, con esta implantación se pueden ajustar los pesos en un intervalo aproximado de -1.0 a 1.0 tal y como se había mencionado anteriormente.

Tomando en cuenta que cada uno de los circuitos de ajuste de peso requiere de un amplificador operacional auxiliar, se decide implementar un sumador con ocho entradas; lo cual es un número razonable si se toma en cuenta que la computadora analógica contará con 10 neuronas. De esta manera, se utilizarán dos DACs para la etapa y se pueden utilizar dos amplificadores cuádruples para la implantación de los pesos.

Entonces se tiene que la salida de la etapa sumadora está dada por

$$V_{OUT} = -\sum_{i=1}^{8} V_i \cdot DAC_i \quad (8),$$

donde

$$DAC_X = \frac{N_D}{128} - \quad (9),$$

Este circuito resulta un alternativa versátil, debido a que todas las entradas poseen pesos que pueden tomar valores positivos y negativos; aunque tiene la desventaja de que la resolución en magnitud se reduce a 7 bits. A pesar de ello, se selecciona este circuito como la mejor alternativa para la implantación de la etapa sumadora debido a la flexibilidad que proporciona.

El circuito para generar pesos positivos y negativos presenta el inconveniente de que la señal viaja por dos rutas distintas. Si tomamos la malla superior de la fig. 5 formada por R1 y R2, se verá que en este caso la señal sólo pasa por el amplificador operacional, mientras que en la malla inferior, la señal debe atravesar también al convertidor digital-analógico añadiendo entonces un retraso adicional a la misma. Esto restringe la posibilidad de utilizar amplificadores operacionales rápidos en el circuito de la fig. 5. Por ello, se utilizaron amplificadores LMC6494, los cuales son cuádruples, tienen un producto ganancia-ancho de banda de 1.5 MHz y un slew rate de 1.3 V/μs. Estas características son similares a las del convertidor MAX505 y, además, el LMC6494 tiene las ventajas de soportar voltajes de entrada de modo común iguales a la fuente de alimentación y presentar a la salida una excursión de fuente a fuente ("input and output rail to rail swing").

Los amplificadores operacionales utilizados como auxiliares en la generación de los pesos deben aceptar niveles de voltaje de entrada de modo común similares a los de polarización. En el circuito de la fig. 5, si se presenta una señal de entrada de 5 Vpp y el código del DAC es tal que la ganancia sea de 0.996 (esto es 255/256), entonces a la salida del DAC tendremos un nivel de voltaje también del orden de 5 Vpp, y debido a la retroalimentación negativa del amplificador operacional auxiliar (LMC6494), éste recibirá un voltaje de modo

común de hasta 5 V. Desde luego debemos recordar que el voltaje de polarización del sistema es de ± 6 V, pero las especificaciones de voltaje de modo común de los amplificadores operacionales típicos no permitirían una operación adecuada de esta etapa a menos que polarizáramos con ± 7 V, lo cual ya estaría en el límite de las especificaciones del DAC MAX505.

Como las señales entre las neuronas de la computadora analógica pueden estar formadas por trenes de pulsos, se prueba experimentalmente la respuesta a un tren de pulsos de la etapa sumadora. Con ello podemos determinar la frecuencia máxima de operación delimitada por el slew-rate. En la fig. 6a se presenta un oscilograma del comportamiento del circuito sumador ante una señal escalón. Se puede observar que el tiempo de estabilización de la respuesta transitoria es de alrededor de 5 µs, lo cual representa el 1 % de la duración de un pulso de una señal cuadrada de 1 kHz, con un ciclo de trabajo del 50 %. Entonces, podemos considerar que la etapa sumadora puede procesar señales cuadradas de hasta 1 kHz con un mínimo de distorsión. En la fig. 6b se muestra la respuesta del sumador a una señal cuadrada de 1.1 kHz, y se puede observar que la distorsión de la señal es poco perceptible. Por otro lado, si aplicamos una señal de entrada senoidal, se obtiene un ancho de banda de 30 kHz. Arriba de esta frecuencia, la forma de onda presenta distorsión debido a limitaciones de slew-rate.

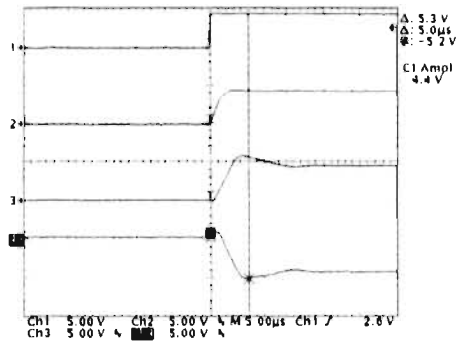


Fig. 6a. Respuesta del circuito sumador a un tren de pulsos. Canal 1: entrada. Canal 2: salida del convertidor digital-analógico. Canal 3: salida del amplificador operacional. Canal 4: salida del sumador.

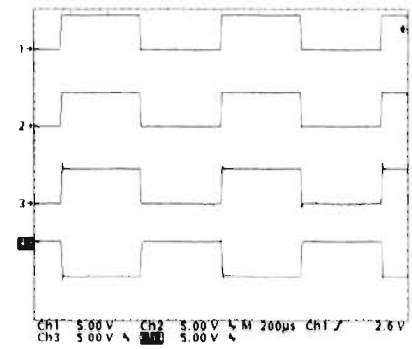


Fig. 6b. Respuesta del circuito sumador a un tren de pulsos (ampliación). Canal 1: entrada. Canal 2: salida del convertidor digital-analógico. Canal 3: salida del amplificador operacional. Canal 4: salida del sumador.

Para poder estimar el error de voltaje de offset que se presenta en la etapa sumadora, se considera el error para código cero y el error a escala completa, el cual es en ambos casos de ±14 mV. A esto habrá que añadírsele el voltaje de offset del amplificador operacional LMC6494 que es de 0.11 mV (típico). El error producido por las corrientes de bias del operacional es despreciable debido a que tiene una I_B=0.15 pA (típica). Entonces el error producido por la etapa de ajuste de peso estará dado por

$$V_{ERR-DC} = E_{DAC} \cdot A_V + V_{OS} \cdot A_N = 28.22mV \qquad (10).$$

donde

E_{DAC} = 14 mV (máximo), es el error a escala completa del convertidor,

A_V = 2, es la ganancia que proporciona el amplificador operacional a la señal proveniente del convertidor,

V_{OS} = 0.11 mV, es el voltaje de offset del amplificador operacional y,

A_N = 2, es la ganancia de ruido del amplificador operacional.

Si consideramos que se van a procesar señales de 10 Vpp y que se tiene una resolución en magnitud de 7 bits, cada paso sería de 78 mV y, por lo tanto, el error en DC de la etapa de ajuste de peso sería menor a la resolución del convertidor.

Después de la etapa de ajuste de pesos se tiene un sumador inversor de 8 entradas (ver figura 7). La ganancia de ruido de este amplificador es de 9, por lo que ese será el factor de amplificación de los errores de offset y de los errores acarreados por las etapas precedentes. Para minimizar estos errores se seleccionó un amplificador operacional OPA27, por tener un bajo voltaje de entrada de offset y bajas corrientes de bias. Como este C.I. tiene un V_{OS}=25μV y una I_B=15nA como valores típicos, y la ganancia de ruido de la configuración mostrada es A_N=9, se tiene que el voltaje de error a la salida debido al voltaje de offset es V_{OO}=225μV y que el voltaje de error a la salida debido a las corrientes de bias es V_{OB}=16μV. Se considera que estos son despreciables en relación a la amplitud de las señales procesadas que es de 10 Vpp.

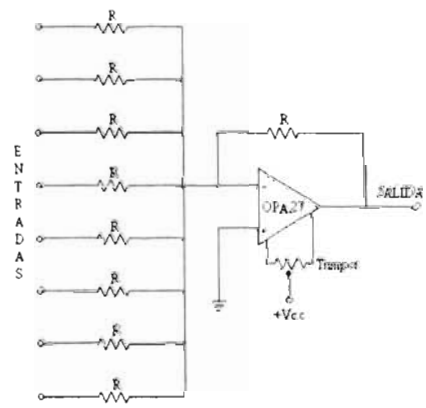


Fig. 7. Sumador inversor.

El amplificador operacional OPA27 cuenta con terminales para el ajuste de offset mediante el empleo de un trimpot. Con esta técnica fue posible reducir los errores de voltaje de offset de toda la etapa sumadora a aproximadamente 7 mV. Para esta última prueba se tomó el peor caso incluso ante diferentes códigos en los DACs.

Por otro lado, el OPA27 tiene un producto ganancia-ancho de banda de 8 MHz típico lo que combinado con la ganancia de ruido A_N=9 nos da un ancho de banda de 888 kHz que es muy cercano al ancho de banda del convertidor analógico-digital (1 MHz). El OPA27 tiene un slew-rate de 1.9 V/μs que es similar a las especificaciones del convertidor digital-analógico MAX505 (SR=3.0 V/μs). En la fig. 8 se presenta un oscilograma donde se puede observar el efecto de ajustar uno de los pesos del sumador cuando se aplica una señal de entrada. En la última curva se puede observar la inversión de fase producto de la aplicación de un peso negativo.

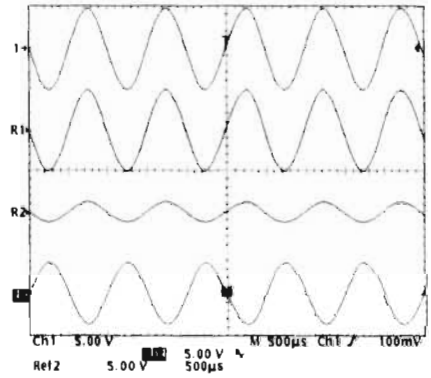


Fig. 8. Cambio de ganancia en el sumador. Canal 1: señal de entrada 9.8 Vpp 1kHz. Canal R1: salida con peso=1.0. Canal R2: salida con peso=0.25. Canal 4: salida con peso= -0.75.

Un ejemplo del funcionamiento del circuito sumador diseñado lo podemos encontrar en la fig. 9. Ahí se observa el efecto en la salida de la suma de dos señales senoidales de entrada de diferente frecuencia.

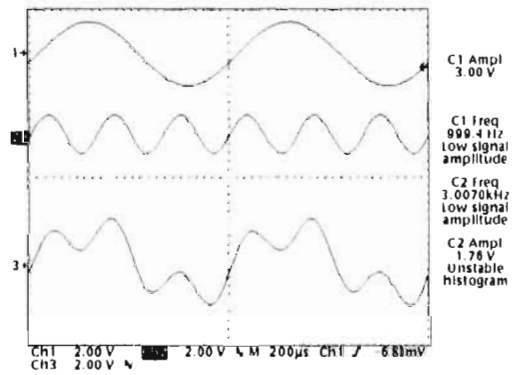


Fig. 9. Circuito sumador. Canal 1: señal de entrada 1 con peso=1. Canal 2: señal de entrada 2 con peso=1. Canal 3: señal de salida.

En la fig. 10 nuevamente se muestra el efecto de las dos señales de entrada, pero ahora con pesos diferentes.

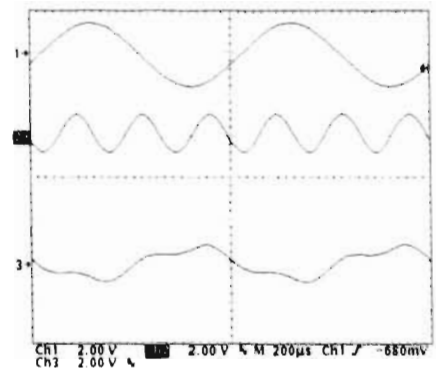


Fig. 10. Circuito sumador. Canal 1: señal de entrada 1 con peso= -0.5. Canal 2: señal de entrada 2 con peso=0.25. Canal 3: señal de salida.

Para comprobar la precisión del circuito de ajuste de peso, se aplicó a una de las entradas un voltaje constante de 5 V y se varió el valor del peso. En la fig.11 se puede ver que los resultados obtenidos son prácticamente iguales a los teóricos. El error máximo que se encontró fue de 30 mV de error absoluto o bien 2% de error relativo.

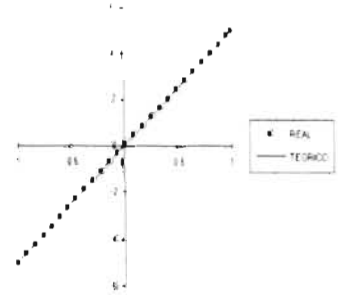
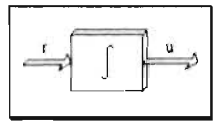


Fig. 11. Gráfica del ajuste de un peso.

Finalmente podemos concluir que se ha diseñado una etapa sumadora de ocho entradas, con pesos positivos y negativos en cada una de ellas, que presenta mínimos errores de offset y puede funcionar con un ancho de banda de 1 kHz.



II.3.2 ETAPA INTEGRADORA

Como se mencionó anteriormente, la etapa integradora tiene la función de añadir un procesamiento temporal de las señales de entrada, de manera similar a como ocurre en la membrana de la neurona biológica. Como modelo de este proceso se utilizarán el integrador con fugas y el integrador y disparo. La implantación electrónica deberá permitir ajustar la constante de integración e incluir un control del potencial de reposo, es decir, el nivel de voltaje que adquiere el integrador en estado estacionario ante una ausencia de señal de entrada.

Algunos estudios en neurofisiología sugieren que el potencial en la membrana de una neurona biológica se puede modelar mediante un integrador. Si además se considera decaimiento del potencial de membrana, el cual intenta representar las fugas de carga que puede presentar la membrana. Tenemos entonces el siguiente modelo

$$\frac{du(t)}{dt} = -\gamma \cdot u(t) + r(t) \qquad (11),$$

donde el término $-\gamma\, u(t)$ corresponde a las pérdidas en la membrana debido a que añade un término de decaimiento exponencial.

En la fig. 12 se muestra el circuito básico propuesto como integrador, el cual se basa en un amplificador operacional de transconductancia (OTA, por sus siglas en inglés). El voltaje de entrada se aplica a la terminal no-inversora del OTA y se produce una corriente de salida que se aplica a un capacitor, el cual sirve de integrador. El voltaje del capacitor se presenta en la salida por medio de un seguidor de voltaje para evitar cargar al circuito del capacitor. El circuito completo genera una función de transferencia de primer orden que se aproxima al modelo matemático del integrador con fugas. La intensidad de corriente de carga del capacitor y, por lo tanto, la constante de tiempo del integrador, depende de la transconductancia g_m del OTA, la cual a su vez depende de la corriente de ajuste I_{BIAS}; por lo tanto, al variar la corriente I_{BIAS} es posible modificar el valor de la constante de tiempo del integrador.

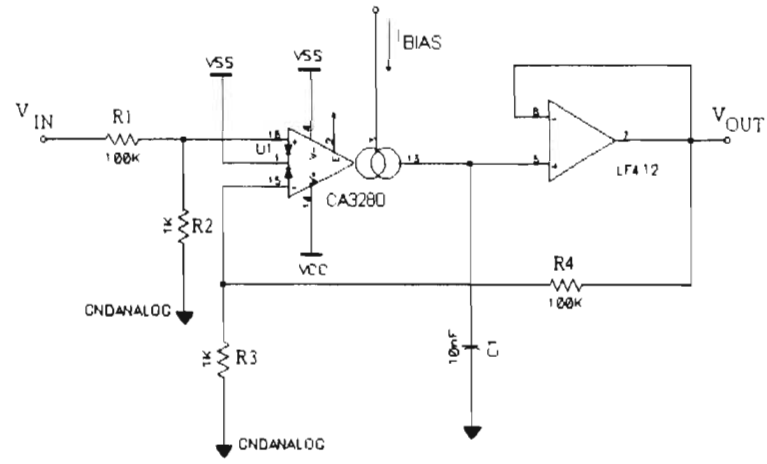


Fig. 12. Esquema básico de un integrador con fugas.

Si consideramos un amplificador operacional de transconductacia ideal podemos obtener el circuito equivalente mostrado en la fig. 13.

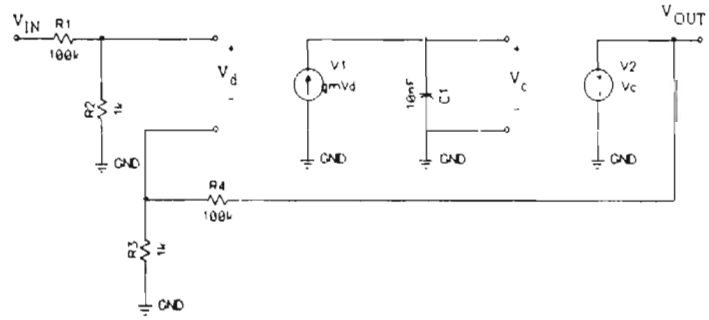


Fig. 13. Modelo del integrador con fugas.

En el circuito podemos observar que el voltaje en el capacitor está dado por

$$V_C = \frac{g_m}{sC} V_d \qquad (12),$$

en donde se tiene que

$$V_d = V_p - V_n \qquad (13),$$

$$V_p = \frac{R_2}{R_1 + R_2} V_{IN} \qquad \text{y} \qquad V_n = \frac{R_3}{R_3 + R_4} V_{OUT} \qquad (14),$$

por lo que sustituyendo (14) en (13) y ésta a su vez en (12) se tiene

$$V_C = \frac{g_m}{sC}\left(\frac{R_2}{R_1+R_2}V_{IN} - \frac{R_3}{R_3+R_4}V_{OUT}\right) \qquad (15).$$

Si consideramos que $R_2 = R_3$ y que $R_1 = R_4$ y como se tiene que $V_{OUT} = V_C$, entonces la función de transferencia del circuito es

$$\frac{V_{OUT}}{V_{IN}} = \frac{1}{\left(\frac{R_1+R_2}{R_2}\right)\frac{C}{g_m}s+1} \qquad (16).$$

donde se observa que la constante de tiempo está dada por

$$\tau = \left(\frac{R_1+R_2}{R_2}\right)\frac{C}{g_m} \qquad (17).$$

la cual depende de la transconductancia del amplificador g_m, misma que se puede variar mediante la corriente I_{BIAS}

$$g_m = \frac{I_{BIAS}}{2V_T} \qquad (18).$$

Considerando las expresiones (17) y (18) tenemos

$$\tau = \left(\frac{R_1+R_2}{R_2}\right)\frac{2V_TC}{I_{BIAS}} \qquad (19),$$

En la parte superior de la fig. 14 se muestra el circuito encargado de fijar la corriente I_{BIAS} del OTA. Consiste en una referencia de voltaje LM336 de 5 V (V_{REF}) conectada a un circuito de ajuste de ganancia similar al utilizado para modificar los pesos de la etapa sumadora, y éste a su vez está conectado a un convertidor de voltaje a corriente. El voltaje V_A a la salida del circuito de ajuste de ganancia está dado por

$$V_A = V_{REF}\left(\frac{N_D}{128} - 1\right) \qquad (20),$$

donde el voltaje V_A podrá ser positivo o negativo dependiendo del valor del código de 8 bits N_D que reciba el DAC. Entonces, el voltaje V_A podrá variar entre -5 y +5 Volts. El convertidor de voltaje a corriente tiene una relación dada por

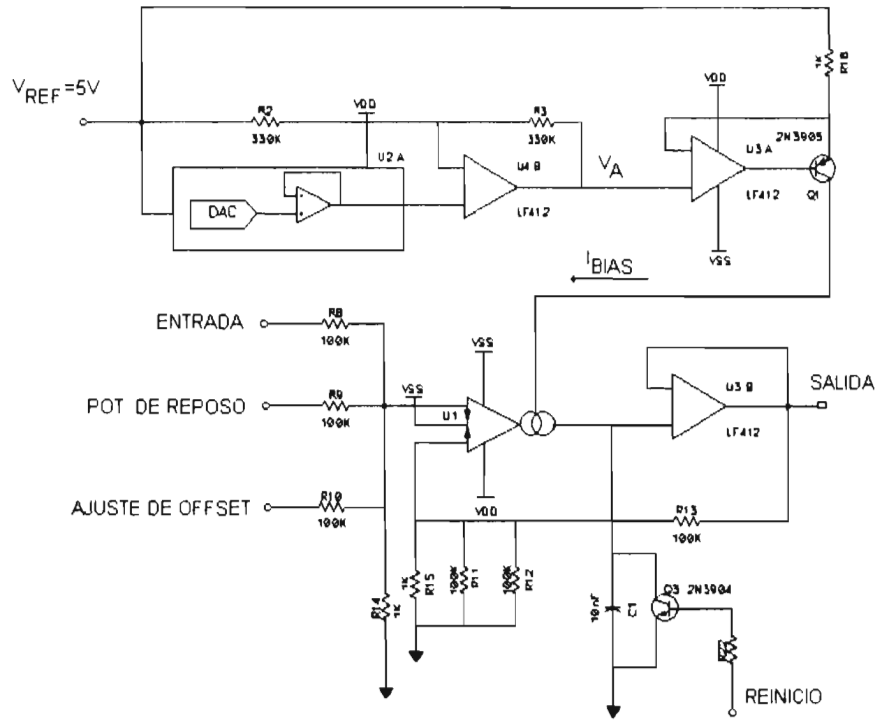


Fig. 14. Diagrama esquemático del circuito integrador.

$$I_{BIAS} = \frac{V_{REF} - V_A}{R_{18}} \qquad (21),$$

y se tiene

$$\tau = \left(\frac{R_A + R_B}{R_B} \right) \frac{256 V_T R_{18} C}{V_{REF}[256 - N_D]} \qquad (22),$$

donde $R_A = R_8 \| R_9 \| R_{10}$ y $R_B = R_{14}$.

De acuerdo con la expresión (23), la constante de tiempo puede variarse en un intervalo de 1 a 256 pasos, sin embargo durante pruebas experimentales sólo se conseguía obtener una variación de 1 a 56. Ello se debía a saturación de la corriente de salida del OTA durante la carga del capacitor. De acuerdo con las hojas de datos, la corriente pico de salida del amplificador (con I_{BIAS}=500 μA) es de 650 μA y con los valores de componentes originalmente considerados se estaba demandando una corriente de salida de hasta 2,200 μA. Para solucionar el problema se ajustó el valor de la corriente de bias mediante la modificación de la resistencia R_{18}, de esta manera se lograron variaciones de la constante de

tiempo de 1 a 200, lo cual se puede considerar suficientemente cercano al intervalo máximo. Por ejemplo, para una cierta calibración, se obtuvieron constantes de tiempo entre 149 μs y 22.4 ms, lo cual resulta adecuado para procesar señales de hasta 10 kHz (ver fig.15).

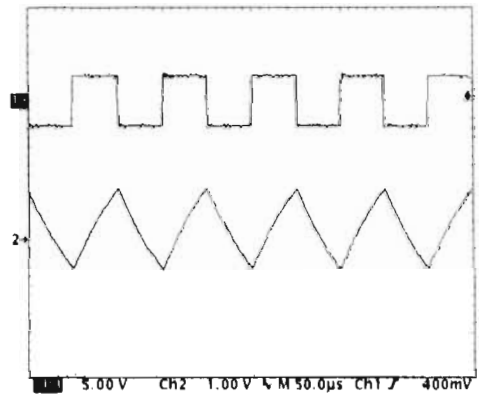


Fig. 15. Salida del integrador ante una señal de entrada cuadrada de 10 kHz, con τ=72 μs.

El integrador así implementado corresponde a un integrador con fugas. Si reescribimos la ecuación (16) tenemos

$$\frac{V_{OUT}}{V_{IN}} = \frac{1}{\tau s + 1} \qquad (23).$$

en forma de ecuación diferencial

$$\tau \frac{dV_{OUT}(t)}{dt} = -V_{OUT}(t) + V_{IN}(t) \qquad (24).$$

la cual corresponde a la ecuación diferencial de un integrador con fugas (ver ecuación 11). Si ahora añadimos el potencial de reposo P el modelo quedará como

$$\tau \frac{dV_{OUT}(t)}{dt} = -V_{OUT}(t) + V_{IN}(t) + P \qquad (25).$$

De esta manera, se puede ver que será suficiente con añadir un voltaje constante a la entrada del integrador para implementar el potencial de reposo. Es por ello que en la fig. 14 se observa además de la terminal de entrada otra adicional para implementar el potencial de reposo. Además, existe otra terminal más para ajustar el offset del integrador. Esta última es necesaria debido a que al ajustar la ganancia del OTA y, por lo tanto, la constante de tiempo del integrador, cambia el voltaje de offset del amplificador. De esta manera será factible calibrar cualquier posible variación. Para lograr estos ajustes se emplean circuitos similares a los utilizados para controlar el peso del sumador, es decir un convertidor digital-analógico con un amplificador operacional adicional para poder implementar ganancias tanto positivas como negativas.

En la fig. 16 se puede observar la respuesta del integrador con fugas a una señal cuadrada de 60 Hz y de 5 V de amplitud, para diferentes constantes de tiempo. El valor de la constante de tiempo se calibró experimentalmente de acuerdo con la expresión (22).

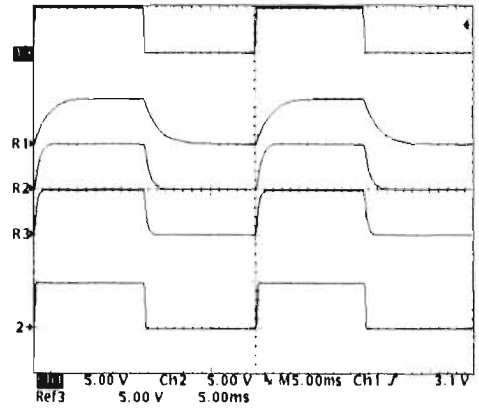


Fig. 16. Respuesta del integrador con fugas. Canal 1: señal de entrada cuadrada 40 Hz. Canal R1: salida del integrador para τ=1.52 ms. Canal R2: salida para τ=0.550 ms. Canal R3: salida para τ=0.284 ms. Canal 2: salida para τ=29μs.

En la fig. 17a se muestra el comportamiento de la salida del integrador ante cambios en el potencial de reposo, el cual se puede ajustar digitalmente. Y en la fig. 17b se puede observar un ejemplo del funcionamiento del integrador y disparo. Una vez que el potencial del integrador alcanza el nivel de umbral fijado (gráfica del canal 2), el integrador genera una señal de disparo representada por las espigas (gráfica del canal 3). Simultáneamente, estas espigas activan el mecanismo que descarga el capacitor del integrador mediante un transistor, el cual se controla mediante la línea de REINICIO (ver fig. 14).

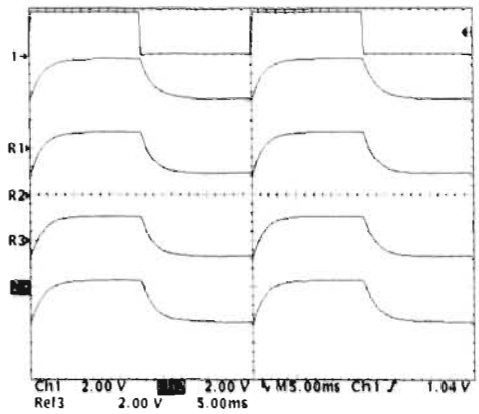


Fig. 17a. Variación del potencial de reposo. Canal 1: señal de entrada cuadrada 40 Hz 1.8V. Canal R1: salida con potencial de reposo=1.992 V. Canal R2: salida con potencial de reposo=0.820 V. Canal R3: salida con potencial de reposo=-0.820 V. Canal 2: salida con potencial de reposo = -1.602 V.

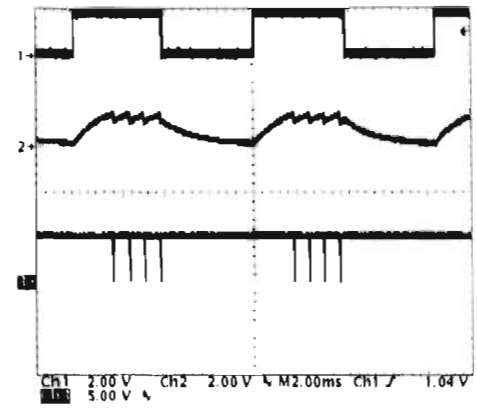
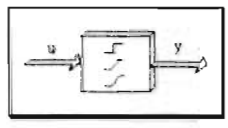


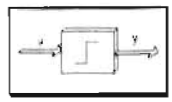
Fig. 17b. Funcionamiento del integrador y disparo. Canal 1: señal de entrada cuadrada 120 Hz 1.8V de amplitud. Canal 2: salida del integrador. Canal 3: señal de disparo

Como se puede ver, ha sido posible implementar un integrador con fugas con opción a funcionar como integrador y disparo, que tiene como características principales el control digital tanto de la constante de tiempo como del potencial de reposo, y es capaz de procesar señales de hasta 10 kHz.



II.3.3 ETAPA DE FUNCIONES DE ACTIVACION

La siguiente etapa de procesamiento que se encuentra en muchos de los modelos de redes neuronales artificiales es una función no-lineal. Con esta función se trata de representar la relación que se presenta entre el nivel de actividad de la neurona biológica (potencial de membrana) y la tasa de disparo. Algunas de las no-linealidades más comúnmente usadas son el escalón, la rampa-saturación y la sigmoide.



II . 3 . 3 . 1 FUNCION ESCALON

La función escalón es la función de activación no-lineal más simple y tiene su origen en la respuesta "todo o nada" de las neuronas biológicas. Los primeros modelos de neuronas consideraron este tipo de función de activación, y en el presente trabajo se propone su implantación con la finalidad de explorar también el comportamiento de redes neuronales de este tipo, ya que la implantación electrónica de este tipo de funciones es bastante simple.

La función escalón se define de la siguiente manera

$$y(u) = \begin{cases} 0 \text{ si } u < \theta \\ 1 \text{ si } u \geq \theta \end{cases} \qquad (26),$$

donde el valor de salida *y(u)* depende de que la entrada *u* supere o no un cierto valor θ de umbral. Una representación gráfica de esta función se muestra en la fig. 18.

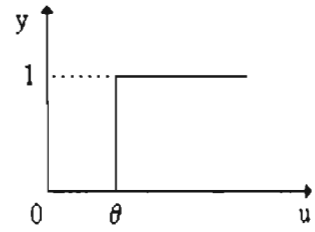


Fig. 18. Función escalón.

Esta función de activación se puede implementar fácilmente utilizando un comparador analógico, tal y como se muestra en la fig. 19.

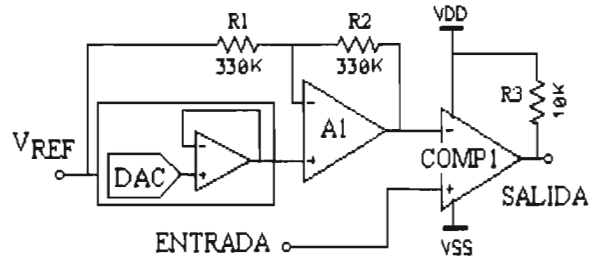


Fig. 19. Implantación electrónica de la función escalón.

En este circuito, se utiliza un comparador LM311 para generar la función escalón. El DAC se utiliza para fijar el voltaje de umbral, para ello se alimenta con un voltaje de referencia de corriente directa C.D., de tal manera que, de acuerdo con la ecuación (6) se tiene

$$\theta = V_{REF}\left(\frac{N_D}{128} - 1\right) \qquad (27).$$

por lo que, de acuerdo con el diagrama de la fig. 19, se tiene que

$$V_{OUT} = \begin{cases} 0 \text{ si } V_{IN} < \theta \\ V_{DD} \text{ si } V_{IN} > \theta \end{cases} \qquad (28),$$

El LM311 tiene un tiempo de respuesta típico de 200 ns, el cual es suficientemente rápido aun para señales de 10 kHz debido a que una señal cuadrada de esta frecuencia tendrá un ancho de pulso de 50 μs. En la fig. 20 se muestra un ejemplo del funcionamiento de la función de activación escalón para formar un integrador y disparo, donde se puede apreciar la velocidad de respuesta del comparador (canal 3).

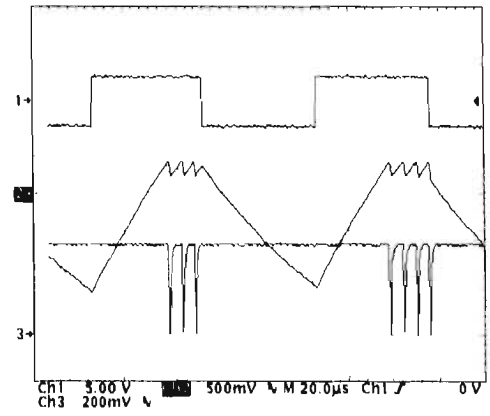
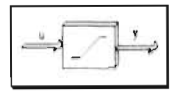


Fig. 20. Función de activación escalón utilizada para implementar un integrador con disparo. Canal 1: entrada. Canal 2: integrador. Canal 3: salida comparador.

La versión final del circuito considera un voltaje de referencia V_{REF} de 5 V, por lo cual el umbral podrá ajustarse dentro del intervalo de -5 a 5 V. Adicionalmente, se tiene contemplada la aplicación de una señal externa de control de umbral mediante el tablero de conexiones; de esta manera, mediante una señal externa es posible generar un comparador cuyo umbral sea dependiente del tiempo.



II.3.3.2 FUNCION RAMPA-SATURACION

La función escalón que se presentó en el punto anterior permite que la salida de la neurona sólo presente dos estados. Para obtener estados de salida intermedios se puede utilizar la rampa-saturación o la tangente hiperbólica. La función de activación Rampa-Saturación se define de la siguiente manera

$$y(u)=\begin{cases} 0 \text{ si } u \le a \\ m(u-a) \text{ si } a < u < \dfrac{\theta}{m} + a \\ \theta \text{ si } u \ge \dfrac{\theta}{m} + a \end{cases} \qquad (29),$$

donde m es la pendiente de la rampa y a es el corrimiento y θ es el nivel de saturación. Su gráfica entrada-salida se muestra en la figura 19.

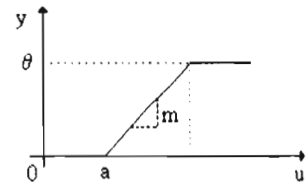


Fig. 19. Función Rampa-Saturación.

El diagrama electrónico del circuito generador de la función rampa-saturación se muestra en la figura 20. Este circuito se encuentra formado por tres etapas: corrimiento, pendiente y saturación. La etapa de corrimiento se encuentra junto a la entrada del circuito y está formada por el DAC1 y el amplificador A1. El objetivo de esta etapa es proporcionar un desplazamiento horizontal a la gráfica entrada-salida según se muestra en la figura 19. Analizando la etapa de corrimiento se tiene

$$V_A = -\frac{R_2}{R_1} V_{IN} - \frac{R_2}{R_3} V_{REF1} + \frac{R_5}{R_4 + R_5}\left(\frac{R_1R_2 + R_2R_3 + R_1R_3}{R_1R_3}\right)\left(\frac{N_D}{256}\right)V_{REF1} \qquad (30),$$

donde si $R_1=R_2=R_3$ y $R_5=2R_4$ se tiene

$$V_A = -V_{IN} + \left(\frac{N_D}{128} - 1\right)V_{REF1} \qquad (31),$$

donde N_D es el código de 8 bits del DAC y V_{REF1}=2.5 V. Por lo tanto, dependiendo de la palabra de control del DAC, se podrán producir corrimientos de ± 2.5 V.

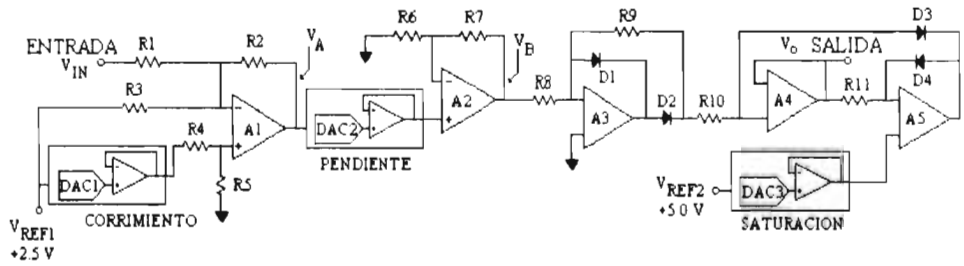


Fig. 20. Implantación electrónica de la función Rampa-Saturación.

La siguiente etapa es la de control de pendiente y está formada por el DAC2 y el amplificador A2. Del análisis de esta etapa se tiene que

$$V_B = \left(\frac{R_6 + R_7}{R_7}\right)\left(\frac{N_D}{256}\right)V_A \qquad (32).$$

si R_7=255R_6 se tiene entonces que

$$V_B = N_D V_A \qquad (33).$$

donde dependiendo del código N_D del DAC2, se tendrá una pendiente ajustable entre 0 y 255 en incrementos unitarios. Como la ganancia proporcionada por A2 es elevada (A_V=256), para minimizar errores se utiliza el amplificador operacional OP37, el cual tiene bajo voltaje de offset (25 μV), baja corriente de bias (15 nA) y un ancho de banda de 63 MHz.

La última etapa es la de saturación. El rectificador de media onda formado por A3 elimina los voltajes negativos y posteriormente el circuito recortador formado por A4 y A5 limita las excursiones positivas a un nivel fijado por el DAC3. Tanto el circuito rectificador como el recortador tienen una configuración tal que evita que la salida de los amplificadores operacionales se sature, lo cual permite una respuesta más rápida.

Algunos ejemplos del funcionamiento de la implantación electrónica de la función rampa-saturación se pueden observar en la figura 21. En ellos se muestra la gráfica entrada-salida del circuito para diferentes valores de pendiente, corrimiento y nivel de saturación. El origen de las gráficas coincide con el ***0***. Se puede observar un pequeño sobretiro en las gráficas, el cual se debe a las limitaciones de Slew-rate (SR) de los amplificadores operacionales utilizados. En este tipo de aplicaciones, el SR es un parámetro relevante en el

desempeño del circuito, debido a que la precisión del recortador depende directamente de la rapidez de respuesta. Para este circuito se utilizaron amplificadores operacionales LF412, los cuales son duales y tienen un SR de 15 V/μs típico y las respuestas que se muestran corresponden a una señal de excitación de 100 Hz. De ser necesario un mayor ancho de banda, deberán cambiarse los amplificadores por otros de mayor SR, por ejemplo el AD827 de Analog Devices, los cuales tienen un SR de 300 V/μs.

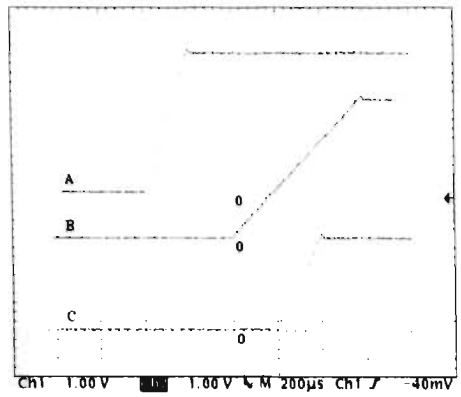


Fig. 21. Respuesta entrada-salida del circuito rampa-saturación. Gráfica A: Pendiente=3, corrimiento=-2 V, saturación= 3V. Gráfica B: Pendiente=1, corrimiento= 0 V, saturación= 3 V. Gráfica C: Pendiente=2, corrimiento= 1V, saturación= 3 V.

El circuito rectificador de media onda que se utilizó es una configuración típica, por lo cual se omite su explicación, y brevemente se describe a continuación el funcionamiento del circuito recortador.

El circuito recortador, que se muestra en la figura 22, funciona de la manera siguiente. Cuando el voltaje de entrada V_{IN} es menor que el voltaje de saturación V_{SAT}, la salida del amplificador A5 se vuelve positiva, lo cual polariza en directa al diodo D4 y polariza en inversa al diodo D3. Por lo tanto, no existe retroalimentación entre A5 y A4, por lo que este último opera únicamente como seguidor de voltaje produciendo que el voltaje de salida sea igual al de entrada ($V_{OUT}=V_{IN}$). Por otro lado, cuando el voltaje de entrada V_{IN} es mayor que el voltaje de saturación V_{SAT}, entonces el voltaje de salida del amplificador A5 se vuelve negativo, polarizando al diodo D4 en inversa y al diodo D3 en directa. De esta manera, se cierra el circuito de retroalimentación entre A4 y A5 a través del diodo D3. Al tender a cero el voltaje diferencial de A5 y debido a su alta impedancia de entrada, se tiene que el voltaje de salida será igual al voltaje de saturación ($V_{OUT}=V_{SAT}$).

desempeño del circuito, debido a que la precisión del recortador depende directamente de la rapidez de respuesta. Para este circuito se utilizaron amplificadores operacionales LF412, los cuales son duales y tienen un SR de 15 V/μs típico y las respuestas que se muestran corresponden a una señal de excitación de 100 Hz. De ser necesario un mayor ancho de banda, deberán cambiarse los amplificadores por otros de mayor SR, por ejemplo el AD827 de Analog Devices, los cuales tienen un SR de 300 V/μs.

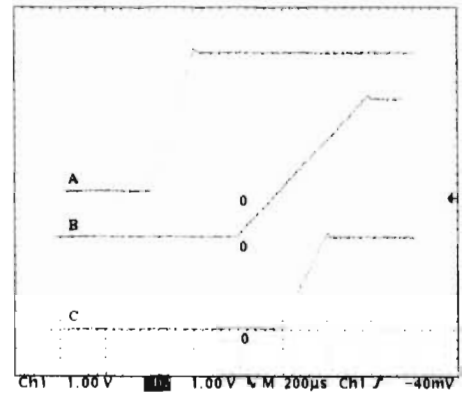


Fig. 21. Respuesta entrada-salida del circuito rampa-saturación. Gráfica A: Pendiente=3, corrimiento=-2 V, saturación= 3V. Gráfica B: Pendiente=1, corrimiento= 0 V, saturación= 3 V. Gráfica C: Pendiente=2, corrimiento= 1V, saturación= 3 V.

El circuito rectificador de media onda que se utilizó es una configuración típica, por lo cual se omite su explicación, y brevemente se describe a continuación el funcionamiento del circuito recortador.

El circuito recortador, que se muestra en la figura 22, funciona de la manera siguiente. Cuando el voltaje de entrada V_{IN} es menor que el voltaje de saturación V_{SAT}, la salida del amplificador A5 se vuelve positiva, lo cual polariza en directa al diodo D4 y polariza en inversa al diodo D3. Por lo tanto, no existe retroalimentación entre A5 y A4, por lo que este último opera únicamente como seguidor de voltaje produciendo que el voltaje de salida sea igual al de entrada ($V_{OUT}=V_{IN}$). Por otro lado, cuando el voltaje de entrada V_{IN} es mayor que el voltaje de saturación V_{SAT}, entonces el voltaje de salida del amplificador A5 se vuelve negativo, polarizando al diodo D4 en inversa y al diodo D3 en directa. De esta manera, se cierra el circuito de retroalimentación entre A4 y A5 a través del diodo D3. Al tender a cero el voltaje diferencial de A5 y debido a su alta impedancia de entrada, se tiene que el voltaje de salida será igual al voltaje de saturación ($V_{OUT}=V_{SAT}$).

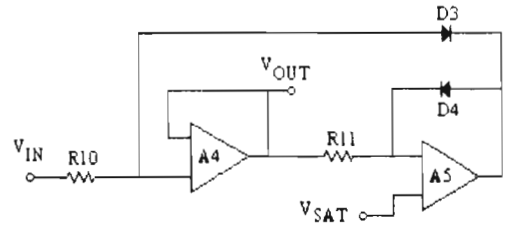


Fig. 22. Circuito recortador.

En la figura 23a se muestra la gráfica de la respuesta entrada-salida del circuito función de activación rampa-saturación, teniendo como entrada una señal senoidal de diferentes frecuencias. La curva A corresponde a una entrada de 100 Hz y no presenta distorsión apreciable. Cuando la entrada tiene una frecuencia de 1 kHz se obtuvo la curva B. En este caso aparece histéresis y se aprecia un pequeño sobretiro en la respuesta. La histéresis se debe al desfasamiento que ocurre entre la entrada y la salida; sin embargo, como se puede apreciar en la figura 23b, el desfasamiento es mínimo. Al medirse experimentalmente se determinó que era menor al 1 %. En la gráfica C de la figura 23a se observa que al aumentar la frecuencia a 3 kHz se presenta un incremento tanto de la histéresis como del sobretiro en la forma de onda de la respuesta. Por lo tanto, se considera que la frecuencia máxima de operación del circuito rampa-saturación es de 1 kHz.

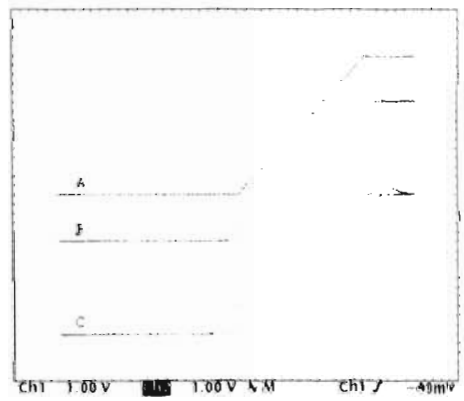


Fig. 23a. Respuesta a diferentes frecuencias del circuito rampa-saturación para una entrada senoidal. A: 100 Hz. B: 1 kHz. C: 3 kHz.

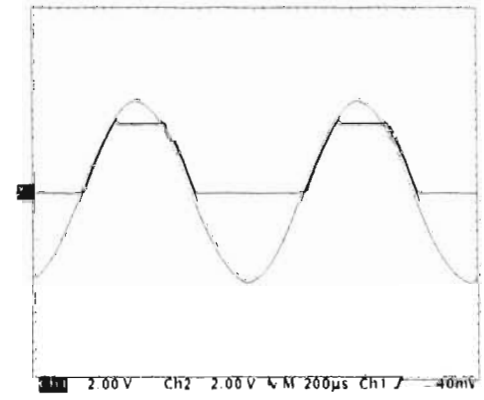
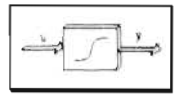


Fig. 23b. Señal de entrada senoidal de 1 kHz y señal de salida del circuito rampa-saturación

Es así como se ha diseñado un circuito función de activación rampa-saturación, cuya pendiente puede ser modificada en el intervalo de 1 a 255. Esto permitirá estudiar la influencia de la pendiente en el comportamiento de redes neuronales, conforme la pendiente se acerca a una función escalón. El circuito permite además el ajuste del nivel de saturación entre 0 y 5 V, y un control del corrimiento entre -2.5 V y +2.5 V.



II.3.3.3 FUNCION TANGENTE HIPERBOLICA

Otra función de activación comúnmente utilizada en los modelos de neuronas es la función de tangente hiperbólica, cuya gráfica se muestra en la figura 24.

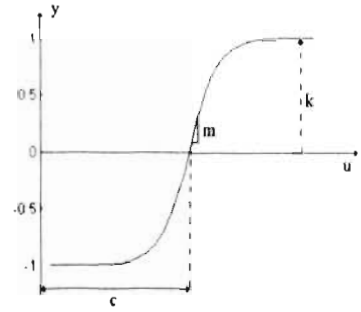


Fig. 24. Tangente hiperbólica.

Para los propósitos de este proyecto se desea controlar los niveles de saturación de la función, la pendiente de la curva y el corrimiento horizontal respecto al origen; por lo tanto, se plantea implementar una función del tipo

$$y(u) = k \cdot \tanh(m \cdot u + c) \qquad (34),$$

donde u es la entrada, m es la pendiente, c es el corrimiento, y k es el factor de saturación.

Para implementar esta función se utiliza un amplificador operacional de transconductancia, el cual tiene una relación entrada salida que se aproxima a una tangente hiperbólica. Se utilizó el OTA CA3280 de Harris (A1), por ser el que menos errores de voltaje de offset presenta, de entre las opciones disponibles. Esto es importante debido a que el voltaje de entrada de offset produce un error a la salida y al cambiar la transconductancia del OTA (al variar su corriente de bias) se produce una variación en el nivel de offset de la señal de salida, lo cual resulta inconveniente para el tipo de procesamiento que se pretende realizar con la neurocomputadora.

En la figura 25 se muestra el circuito básico para genera la tangente hiperbólica. Las resistencias R1 y R2 forman un divisor de voltaje que sirven para escalar el voltaje de entrada al OTA. El circuito CA3280 cuenta con unos diodos de linealización de la respuesta del amplificador; sin embargo, como en este caso nos interesa aprovechar la característica no-lineal del amplificador, en el circuito mostrado estos diodos se encuentran desactivados mediante una conexión a VEE. La resistencia R3 sirve para transformar la corriente de salida del amplificador de transconductancia en una señal de voltaje. Para evitar carga a la salida del OTA, utilizamos un amplificador operacional en configuración de seguidor de

voltaje (A2) dado que el circuito CA3280 no cuenta con un buffer de salida como lo tendría un LM13600 (amplificador operacional de transconductancia de National Semiconductor).

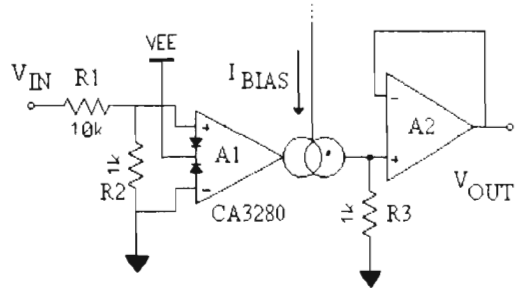


Fig. 25. Generación de la tangente hiperbólica.

La implantación de la tangente hiperbólica queda complementada con una etapa de ajuste de pendiente y de corrimiento, más otra etapa de ajuste de ganancia (nivel de saturación). según se muestra en el diagrama de bloques de la figura 26.

Fig. 26. Diagrama de bloques de la función de activación tangente hiperbólica.

El circuito utilizado para variar la pendiente y el corrimiento de la tangente hiperbólica se muestra en la figura 27. Se utiliza el DAC1 y el amplificador A1 para producir un nivel de corrimiento entre -2.5 V y +2.5 V (ver punto II.3.1 fig. 5). Este nivel de corrimiento se suma al voltaje de entrada por medio del amplificador operacional A2. Posteriormente, el DAC2 permite ajustar la pendiente, ya que es utilizado como divisor de voltaje controlado digitalmente.

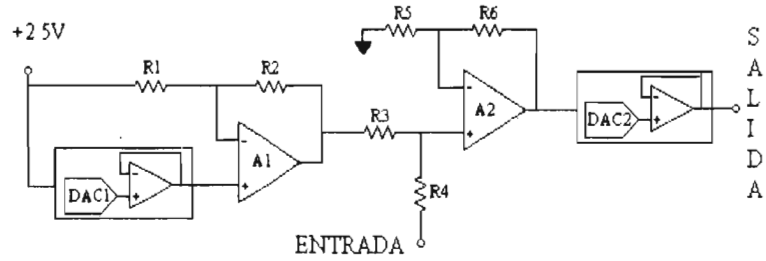


Fig. 27. Circuito de ajuste de pendiente y corrimiento.

Debido a los errores internos del OTA, la tangente hiperbólica que se produce puede estar descentrada con respecto al origen. Esto es, puede estar corrida vertical u horizontalmente. Para permitir que la curva esté centrada en el origen, se han añadido dos circuitos: uno para ajuste horizontal, y otro para ajuste vertical. El circuito de ajuste horizontal (figura 28) se coloca entre la etapa de ajuste de pendiente-corrimiento y el OTA. El circuito de ajuste vertical es idéntico al de ajuste horizontal pero se coloca a la salida del OTA.

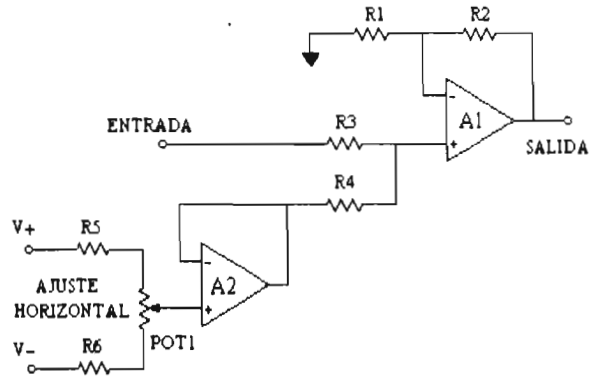


Fig. 28. Circuito de ajuste de posición horizontal.

En la figura 29 se muestra el circuito de ajuste de la corriente de bias del OTA. Este circuito sirve para calibrar la pendiente de la tangente hiperbólica, ya que ésta depende de la transconductancia del OTA.

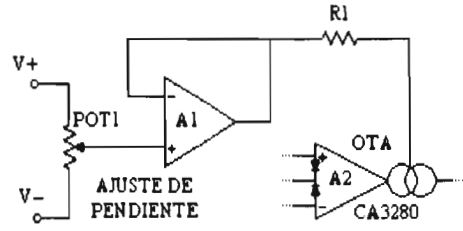


Fig. 29. Ajuste de la corriente de bias del OTA.

Como se mencionó anteriormente el amplificador operacional de transconductancia tiene una relación entrada-salida que se aproxima a una tangente hiperbólica. En la figura 30 se muestra un diagrama básico de un OTA. En él se observa el par diferencial formado por Q1 y Q2 y cuatro espejos de corriente, con ellos se produce una corriente de salida I_o igual a la diferencia de las corrientes de colector de los transistores del par diferencial (I_b-I_a).

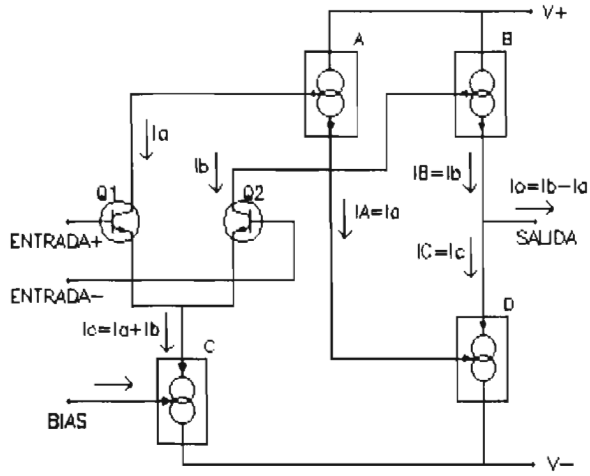


Fig. 30. Diagrama básico de un OTA.

Analizando el circuito para señal grande tenemos que las corrientes de colector de los transistores Q1 y Q2 están dadas por

$$I_a = I_S e^{\kappa(V_1 - V_L)} \quad (35).$$

$$I_b = I_S e^{\kappa(V_2 - V_L)} \quad (36).$$

donde $\kappa = \frac{1}{V_T}$, siendo V_T el voltaje térmico. La corriente I_c drenada por la fuente de corriente C está entonces dada por

$$I_c = I_{E1} + I_{E2} = \frac{I_{C1} + I_{C2}}{\alpha_F} = \frac{I_a + I_b}{\alpha_F} \quad (37),$$

donde, si consideramos que Q1=Q2, podemos escribir

$$\alpha_F I_C = I_S e^{\kappa(V_1 - V_L)} + I_S e^{\kappa(V_2 - V_L)} = I_S e^{-\kappa V_L}(e^{\kappa V_1} + e^{\kappa V_2}) \quad (38).$$

Despejando el término $e^{-\kappa V_L}$, tenemos

$$e^{-\kappa V_L} = \frac{\alpha_F I_c}{I_S(e^{\kappa V_1} + e^{\kappa V_2})} \quad (39).$$

Sustituyendo (39) en (35) obtenemos

$$I_a = e^{\kappa V_1} e^{-\kappa V_L} = \frac{\alpha_F I_c e^{\kappa V_1}}{I_S(e^{\kappa V_1} + e^{\kappa V_2})} \quad (40).$$

Procediendo de manera similar, se puede despejar I_b, obteniéndose

$$I_b = e^{\kappa V_2} e^{-\kappa V_1} = \frac{\alpha_F I_c e^{\kappa V_2}}{I_S (e^{\kappa V_1} + e^{\kappa V_2})} \qquad (41).$$

De las expresiones para I_a e I_b se tiene que

$$I_b - I_a = \frac{\alpha_F I_c (e^{\kappa V_2} - e^{\kappa V_1})}{I_S (e^{\kappa V_1} + e^{\kappa V_2})} \qquad (42),$$

lo cual se puede expresar en la forma

$$I_b - I_a = \frac{\alpha_F I_c}{I_S} \frac{(e^{\kappa V_2} - e^{\kappa V_1})}{(e^{\kappa V_1} + e^{\kappa V_2})} \frac{e^{\frac{-\kappa V_1}{2}} e^{\frac{-\kappa V_2}{2}}}{e^{\frac{-\kappa V_1}{2}} e^{\frac{-\kappa V_2}{2}}},$$

$$I_b - I_a = \frac{\alpha_F I_c}{I_S} \frac{e^{\left(\frac{\kappa V_2}{2} - \frac{\kappa V_1}{2}\right)} - e^{\left(\frac{\kappa V_1}{2} - \frac{\kappa V_2}{2}\right)}}{e^{\left(\frac{\kappa V_1}{2} - \frac{\kappa V_2}{2}\right)} + e^{\left(\frac{\kappa V_2}{2} - \frac{\kappa V_1}{2}\right)}} = \frac{\alpha_F I_c}{I_S} \frac{e^{\frac{\kappa}{2}(V_2 - V_1)} - e^{\frac{-\kappa}{2}(V_2 - V_1)}}{e^{\frac{\kappa}{2}(V_2 - V_1)} + e^{\frac{-\kappa}{2}(V_2 - V_1)}},$$

de donde finalmente se obtiene que

$$I_b - I_a = \frac{\alpha_F I_c}{I_S} \tanh \frac{\kappa (V_2 - V_1)}{2}.$$

Puesto que $I_o = I_b - I_a$, que $I_{BIAS} = I_c$ y si consideramos $\alpha_F \approx 1$ tenemos finalmente

$$I_o = \frac{I_{BIAS}}{I_S} \tanh \frac{\kappa (V_2 - V_1)}{2} \qquad (43),$$

En la práctica, los transistores del par diferencial de entrada no son iguales, lo que dará lugar a errores de voltaje de offset; además, los espejos de corriente no son ideales, lo que también generará errores.

Finalmente, el bloque de ganancia de esta etapa se muestra en la figura 31. El circuito formado con los amplificadores A1 y A2 permite ajustar la posición vertical de la curva, mientras que el DAC1 permite fijar la ganancia, es decir el nivel de saturación de la función. En este caso, el DAC funciona como atenuador controlado digitalmente.

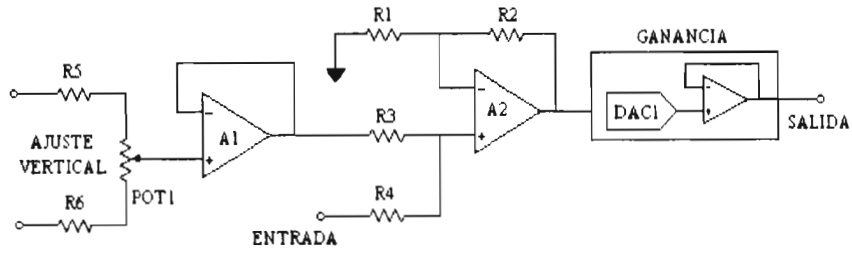


Fig. 31. Circuito de control de ganancia y ajuste vertical.

En las figuras 32a y 32b se muestran dos oscilogramas de la gráfica entrada-salida que se obtiene del circuito función de activación tangente hiperbólica. El origen se encuentra en el centro del oscilograma. Como puede verse, se puede modificar fácilmente y con un grado aceptable de precisión los parámetros de la función. La diferencia entre tener una u otra respuesta consiste en mover los tres parámetros desde la computadora personal y esta operación se realiza mediante movimientos del ratón ("mouse").

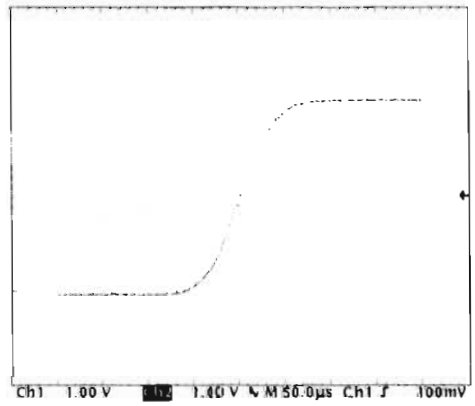


Figure 32a. Tangente hiperbólica. Pendiente=2.5, corrimiento=0V, ganancia (nivel de saturación)=2V.

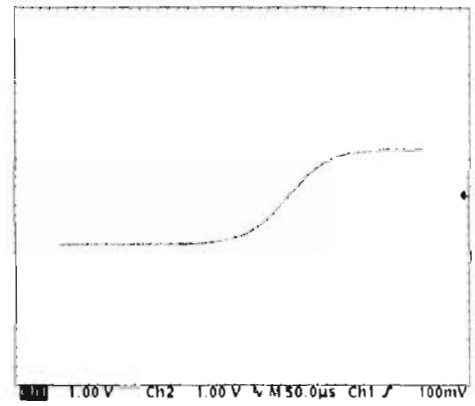


Figure 32b. Tangente hiperbólica. Pendiente=1.0, corrimiento=1V, ganancia (nivel de saturación)=1V.

En la figura 33a se ilustra la gráfica entrada-salida del circuito tangente hiperbólica para una entrada senoidal de diferentes frecuencias. La gráfica A corresponde a una señal de entrada de 100 Hz. Se puede observar que a esta frecuencia no existe distorsión aparente. En la gráfica B la señal de entrada es de 1 kHz. En este caso aparece histéresis en la curva, pero ésta es debida al desfasamiento que ocurre en el circuito. Este desfasamiento, sin embargo, es mínimo según se puede apreciar en la figura 33b, y experimentalmente resulta menor al 1%. En la gráfica C de la figura 33a se presenta el comportamiento con una entrada de 10 kHz. El desfasamiento es ahora mayor al 3 % y eso se ve reflejado en un incremento de la histéresis. Finalmente podemos decir que el circuito opera adecuadamente hasta frecuencias de 1 kHz.

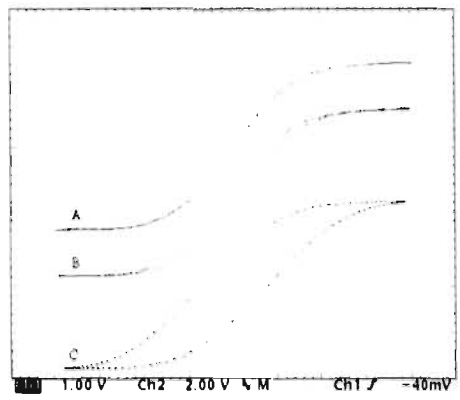


Fig. 33a. Respuesta entrada-salida del circuito tangente hiperbólica ante una entrada senoidal. A: 100 Hz. B: 1 kHz. C: 10 kHz.

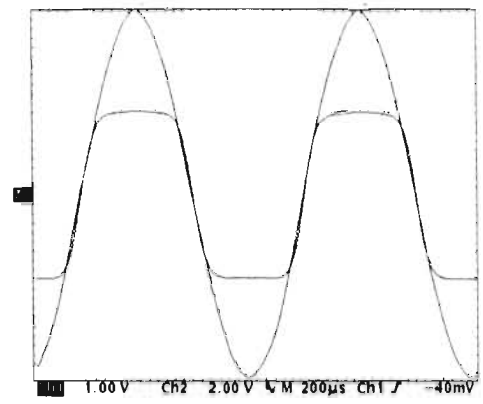
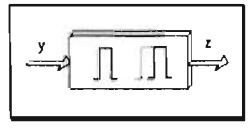


Fig. 33b. Señal de entrada senoidal de 1 kHz y señal de salida del circuito tangente hiperbólica.

Es así como se ha diseñado un circuito función tangente hiperbólica cuya pendiente se puede variar en el intervalo de 0 a 5, con un nivel de saturación que se puede ajustar entre 0 y 4 V, y con un corrimiento entre -2.5 y 2.5 V.



II . 3 . 4 ETAPA GENERADORA DE PULSOS

El objetivo de esta etapa es producir un disparo o tren de disparos, dado que representan una posible forma de transmisión de información. Se cree que la información en las redes neuronales biológicas depende de la frecuencia de los pulsos y no de la forma de ellos; por lo tanto, optamos por simplificar la generación de pulsos y producir únicamente pulsos rectangulares de duración ajustable. Adicionalmente, y como una forma de replicar el retardo que sufren los impulsos nerviosos al viajar por el axón, hemos implementado un circuito de retardo mediante el cual se puede generar el pulso un cierto intervalo de tiempo después de recibida la señal de disparo.

Una forma muy práctica de generar pulsos rectangulares es mediante un circuito monoestable basado en el circuito integrado LM555. Para controlar la duración del pulso generado es necesario modificar los valores de los componentes externos al LM555, o bien, aplicar un voltaje a la terminal de control (CON). Esta última opción resulta en un control no-lineal del ancho del pulso debido a la respuesta exponencial del circuito RC externo que se utiliza con el LM555. Para lograr una respuesta lineal se optó por cargar el capacitor de referencia mediante una fuente de corriente constante tal y como se muestra en la figura 34.

El generador de pulsos funciona de la siguiente manera (ver figura 34). Cuando el voltaje de entrada supera un nivel de referencia fijado mediante el DAC, el comparador COMP2 genera una señal de disparo baja. Esta provoca que la salida del temporizador LM555 cambie a alto y se inicia la carga del capacitor C1. Hasta este momento, el capacitor se mantenía en cero debido a que está conectado al circuito de descarga del LM555. Mediante el potenciómetro R1 se fija un voltaje de control positivo V_A, el cual sirve de entrada a una fuente de corriente constante formada por A1 y Q1 de tal manera que

$$I_{C1} = \frac{V_{DD} - V_A}{R_2} \qquad (44).$$

Con esta corriente se carga el capacitor C1 de tal manera que

$$V_{C1} = \frac{I_{C1} \cdot \Delta t}{C_1} \qquad (45),$$

y el voltaje del capacitor crecerá en forma de rampa como se muestra en la figura 35. Cuando el voltaje en el capacitor alcanza al voltaje de umbral (V_{TH}), el comparador COMP1 manda una señal de restablecimiento al LM555, la cual hace que la salida del LM555 vuelva a un estado bajo, y además activa el circuito de descarga (DIS) con el que se regresa al capacitor C1 a condiciones iniciales nulas.

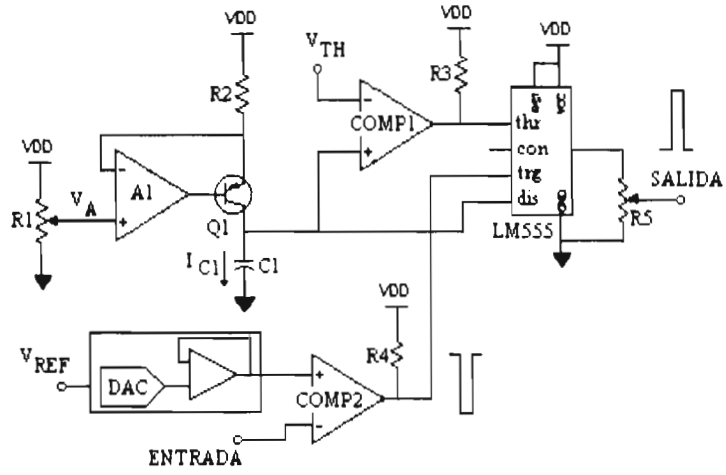


Fig. 34. Generador de pulsos.

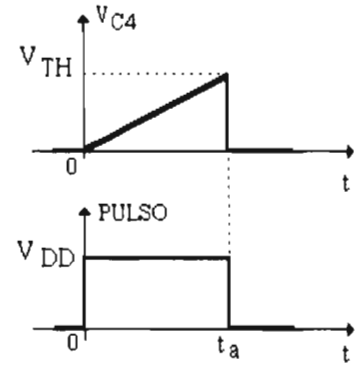


Fig. 35. Voltaje en el capacitor C4.

De (45) y (46) tenemos que el ancho del pulso así producido estará dado por

$$\Delta t = \frac{V_{TH} C_1 R_2}{V_{DD} - V_A}, \tag{46}$$

El voltaje de umbral se controla digitalmente por medio del circuito de la figura 36, en el cual se utiliza una referencia de voltaje integrada (LM336) y un DAC para generar el voltaje V_{TH}. El voltaje V_A se utiliza para calibrar la constante de integración. Finalmente, el potenciómetro R5 se utiliza para fijar la amplitud del pulso generado por el temporizador LM555. Esto se requiere debido a que el voltaje que produce el LM555 no es igual al voltaje de polarización sino ligeramente menor; con esta alternativa es posible igualar el voltaje de salida de todos los temporizadores utilizados en el sistema.

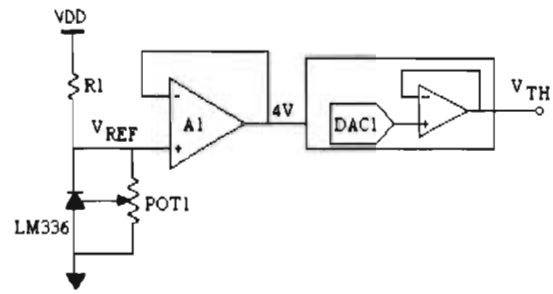


Fig. 36. Control del ancho de pulso mediante un voltaje de umbral.

Una vez que se genera el pulso, se puede aplicar esta señal a un circuito de retardo. El objetivo de esta etapa es generar un pulso idéntico pero con un retardo en el tiempo. Para ello se emplea el circuito de la figura 37. Este circuito funciona de la siguiente manera. El comparador COMP1 recibe el pulso de salida de la etapa generadora de disparo (pulso) y lo invierte para con ello activar el temporizador LM555. La salida de éste es un voltaje alto, el cual se aplica al potenciómetro R4 para fijar el voltaje Vx con el que se controla una fuente

de corriente constante formada por A1 y Q1. La corriente que se hace pasar por el capacitor C1 genera una rampa de voltaje como la mostrada anteriormente en la figura 35. Cuando el voltaje en el capacitor C1 supera el voltaje de umbral V_{TH}, entonces a la salida (COMP2) se produce un voltaje bajo que se canaliza a otra etapa generadora de pulso similar a la mostrada en la figura 34. El resultado final es que se produce otro pulso pero retardado, tal y como se había propuesto.

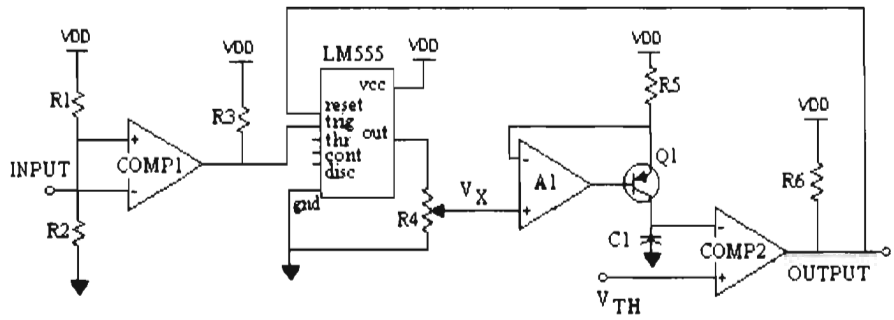


Fig. 37 . Circuito de retardo.

En la figura 38 se muestra un ejemplo de la respuesta del generador de pulsos ante una señal de entrada cuadrada (canal 1). El flanco positivo activa la generación de un pulso de salida de 15 ms de duración (canal 2), mismo que se puede reproducir mediante el circuito de retardo, a los 8 ms de la señal de disparo (canal 3). Es decir, el circuito genera un pulso sincronizado con una señal de disparo, pudiendo ser generado de manera instantánea o de manera retardada con respecto a la señal de disparo.

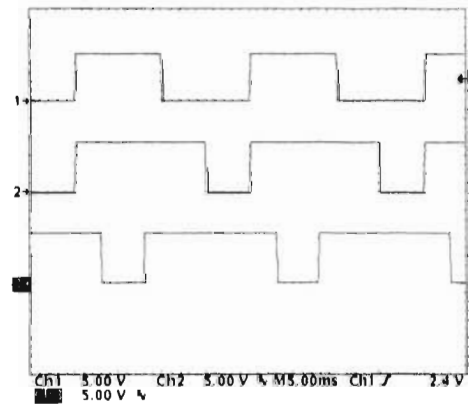


Figure 38. Oscilograma del generador de pulso. Canal 1: señal de entrada. Canal 2: pulso de salida (ancho del pulso=15.0 ms). Canal 3: pulso de salida retardado (ancho del pulso= 15.0 ms y retardo=8.0 ms).

El circuito desarrollado puede generar pulsos con una duración entre 0.1 ms y 25 ms, y además puede introducir un retardo entre 0.2 ms y 50 ms. En la figura 39a se observa como

es posible retardar un tren de pulsos, siempre que la duración del retardo sea menor a la duración del ancho del pulso. Por otro lado, pulsos aislados se pueden retrasar hasta 50 ms según se muestra en la figura 39b.

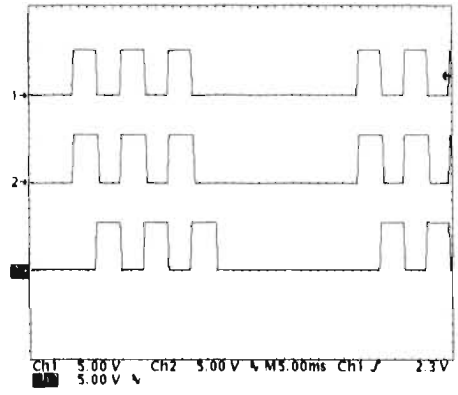


Fig. 39a. Retraso de un tren de pulsos. Canal 1: Señal de entrada. Canal 2: Pulsos de salida. Canal 3: Salida retardada.

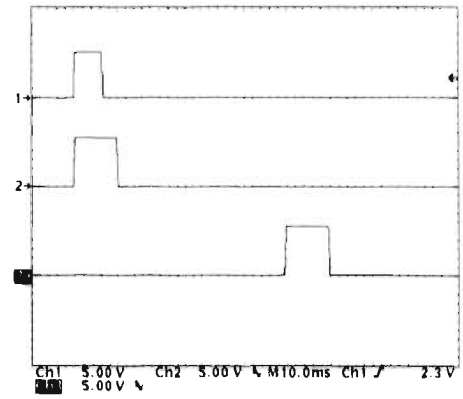


Fig. 39b. Retraso de un pulso aislado. Canal 1: Señal de entrada. Canal 2: Pulso de salida. Canal 3: Pulso retardado 50 ms.

De esta manera se ha presentado el desarrollo del modelo electrónico de neurona, el cual está formado por cuatro etapas que en conjunto presentan 20 parámetros de ajuste y 7 opciones de función cada una de las cuales es controlada desde la computadora personal.

II . 4 INTERFAZ CON LA COMPUTADORA PERSONAL

Para servir de enlace entre la computadora analógica y la computadora personal, se desarrolló una interfaz basada en microcontrolador. La interfaz se encarga de establecer la comunicación serial con la computadora personal para recibir los valores de todos los parámetros de ajuste del sistema que deberán ser cargados en los DACs. Esta interfaz se basa en un microcontrolador PIC16C74 del fabricante Microchip.

En la figura 40 se muestra el diagrama de bloques de la interfaz, donde se pueden observar sus componentes principales: el microcontrolador (PIC16C74), un manejador de comunicación serial (MAX232) y un conector paralelo para comandar la lógica de decodificación y los DACs que se encuentran en la computadora analógica..

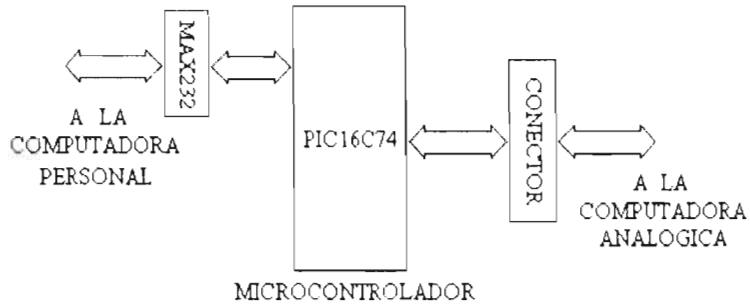


Fig. 40. Diagrama de bloques de la interfaz con la computadora personal.

El C. I. PIC16C74 es un microcontrolador de 8 bits tipo RISC (Computadora con conjunto reducido de instrucciones, por sus siglas en inglés). Cuenta con un puerto de comunicaciones seriales, 33 pines de entrada/salida y 4Kx14 bits de memoria EPROM (Memoria de sólo lectura programable y borrable, por sus siglas en inglés), entre otras prestaciones. Además, poniéndolo a operar con un reloj de 20 MHz, ejecuta una instrucción en 200 ns.

Las tareas que se realizan con el microcontrolador consisten esencialmente en recibir datos por el puerto serial y enviarlos a un puerto paralelo junto con una dirección para identificar al DAC o Latch requerido. Para realizar esta operación se pudo haber seleccionado cualquier microcontrolador; sin embargo, se seleccionó el PIC16C74 por su facilidad de programación, por encontrarse fácilmente en el mercado y por que cuenta con suficiente memoria para futuras ampliaciones al sistema.

El microcontrolador cuenta, como se mencionó anteriormente, con 33 pines de entrada/salida agrupados en cinco puertos según se muestra en la figura 41. Los pines RC6 y RC7 del puerto C se utilizan para la comunicación serial ya que tienen como función alterna las líneas del puerto USART ("universal synchronous asynchronous receiver transmiter"). Los pines RC0-RC2 se utilizan para conectar tres LEDs que indicarán el estado del sistema. Un LED verde indicará que el sistema está listo, uno amarillo que se está transmitiendo información entre la computadora analógica y la computadora personal. El tercer LED queda disponible para futuras modificaciones al sistema.

La asignación anterior de puertos, deja al sistema con dos puertos de 8 bits disponibles (el puerto B y el D). Se decide utilizar al puerto D para el envío de datos y el puerto B para el envío de direcciones, lo cual permite direccionar 256 localidades. Adicionalmente se utilizan las líneas RA0, RA1, RE0 y RE1 para generar las líneas adicionales, A0, A1, LD y WR respectivamente, que requieren los DACs.

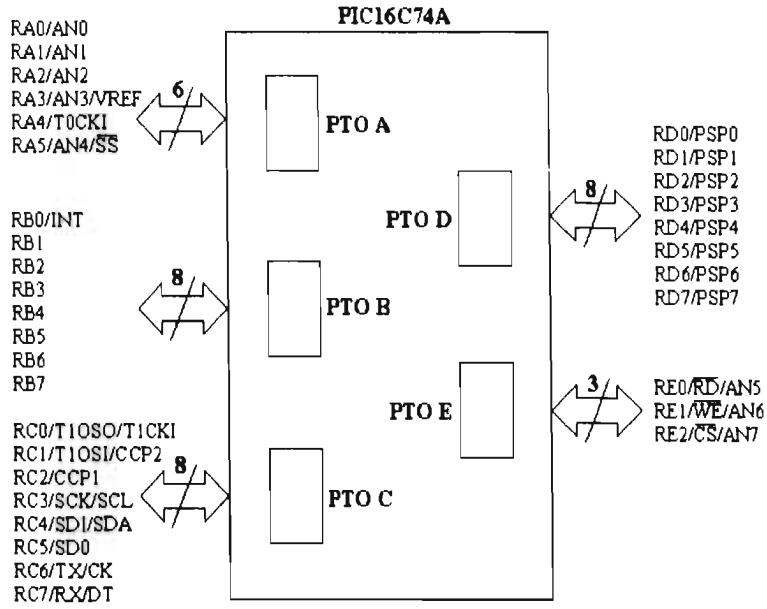


Fig. 41. Puertos del microcontrolador.

De manera resumida el programa del microcontrolador efectúa las siguientes tareas (ver fig. 42). En primer lugar inicializa los registros que habilitan la comunicación serial a una velocidad de 2400 bauds. Después se ejecuta una rutina que espera la llegada de un comando por el puerto serial. Una vez que se recibe el comando, se lee en una tabla la dirección de la subrutina de atención a ese comando. La subrutina se encarga entonces de recibir el resto de la información y enviarla al DAC correspondiente.

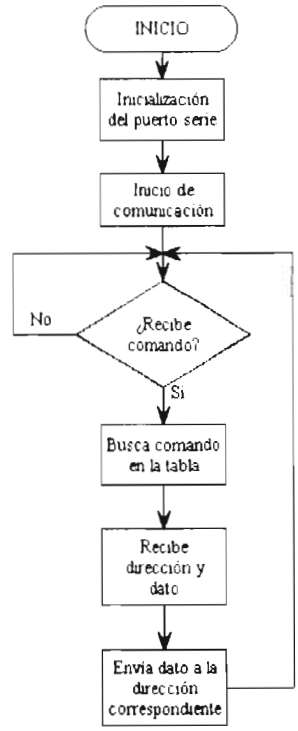


Fig. 42. Diagrama de flujo del programa de la interfaz.

En la figura 43 se puede observar el diagrama esquemático de la tarjeta de interfaz. En él se muestra el conector CON1, el cual se conecta con las tarjetas base de cada una de las diez neuronas. Estas tarjetas base representan una carga equivalente a una compuerta LS-TTL o una compuerta CMOS, es decir, a una corriente de 200 μA_{MAX} (ver tabla 1). Si multiplicamos esta carga por el número de tarjetas base (10), tendremos la corriente máxima que será demandada a los pines del microcontrolador. En este caso la corriente será de 2 mA, la cual es inferior a la capacidad de 25 mA que tienen nominalmente. Adicionalmente se encuentran conectados tres LEDs directamente a los pines del microcontrolador; sin embargo, estos representan una carga de 9 mA que también es inferior a la capacidad del microcontrolador. Estos LEDs se utilizan para indicar el modo de operación de la interfaz: verde indica que el sistema está listo, amarillo que se están transfiriendo datos y rojo que el sistema analógico está activo, es decir, que la computadora analógica se encuentra procesando.

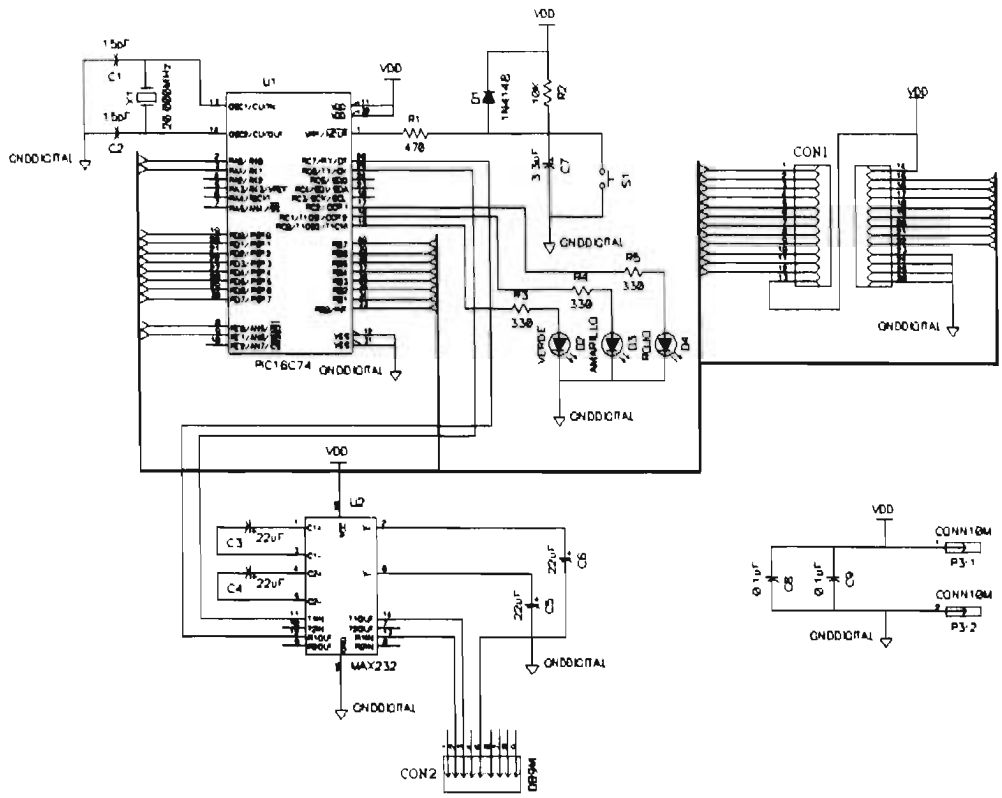


Fig. 43. Diagrama esquemático de la interfaz.

Carga	Corriente
1 LS-TTL 74LS244	I_{IH}=20 μA I_{IL}=200μA
1 CMOS GAL	I_{IH}=10 μA I_{IL}=150 μA

Tabla 1. Carga de la interfaz.

En la figura 44 se muestra el diagrama esquemático de la tarjeta base. En ella se encuentran la lógica de decodificación y unos buffers que refuerzan el bus de datos, las líneas de direcciones y la línea de carga. La lógica de decodificación está implementada por medio de un GAL y la carga que se le conecta está constituida por un DAC o un Latch (ver tabla 2). por lo que la carga máxima que se presenta será de 400 μA que está muy por debajo de los 16 mA que puede proporcionar el GAL. Los buffers tienen una carga de 6 DACs y 2 Latches (ver tabla 2), lo cual constituye una carga máxima de 836 μA que es inferior al límite proporcionado por el buffer que es de 24 mA.

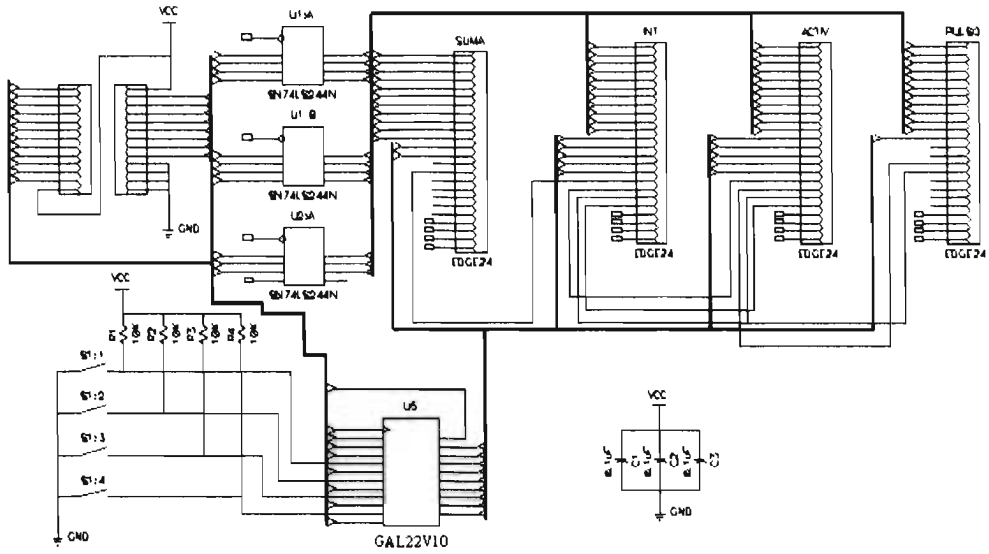


Fig. 44. Diagrama esquemático de la tarjeta base.

Carga	Corriente
1 CMOS MAX505	I_{IH}=1 μA I_{IL}=1 μA
1 LS-TTL 74LS573	I_{IH}=20 μA I_{IL}=400 μA

Tabla 2. Carga de la lógica de decodificación y de los buffers.

En la figura 45 se ilustra el diagrama esquemático que se implementó en el GAL. Las compuertas XOR y la compuerta NAND funcionan como un comparador de 4 bits, el cual se utiliza para identificar la dirección de la neurona. Las entradas A0-A3 serán las líneas proporcionadas por el microcontrolador y se compararán con las entradas B0-B3 que están fijadas mediante un DIP switch. De esta manera, cada tarjeta base puede tener una dirección diferente que se puede establecer mediante el DIP switch, existen 16 posibles combinaciones. La dirección se completa con las líneas SEL0-SEL2 que una vez decodificadas generan las líneas de selección para seis DACs (BWR0-BWR5) y dos Latches (LATCH1-LATCH2). Finalmente la línea SEL7 puede duplicar el número posible de direcciones, lo cual permitiría manejar hasta 32 neuronas, o bien puede permitir el direccionamiento de dispositivos adicionales que se colocaran en versiones ampliadas de alguna de las etapas del modelo. La generación del archivo de programación del GAL se realizó utilizando la utilería PLSYM del programa PSPICE de Microsim, a partir de la definición del diagrama esquemático.

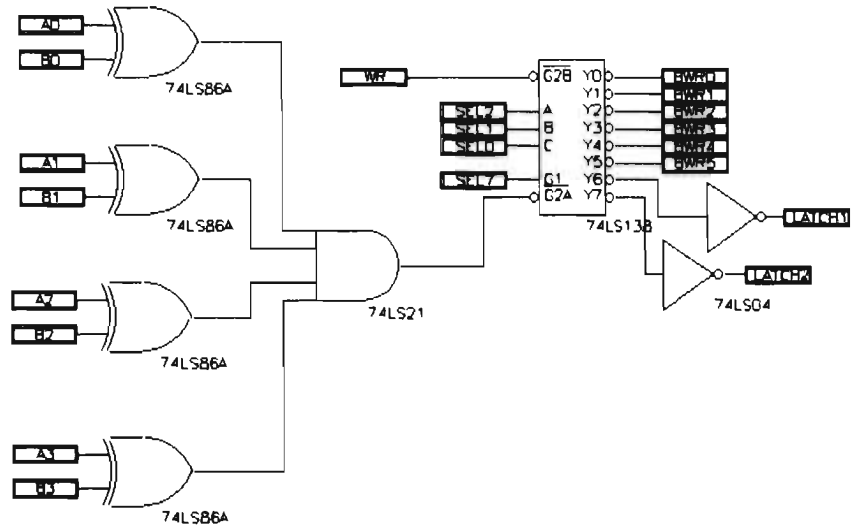


Fig. 45. Diagrama esquemático de la lógica de decodificación implantada en el GAL.

Las líneas SEL0-SEL7 provienen del puerto B del microcontrolador (líneas RB0-RB7 respectivamente). Los bits menos significativos (RB0-RB2) determinan el dispositivo a direccionar dentro de cada neurona de acuerdo con lo mostrado en la tabla 3. Los siguientes cuatro bits (RB3-RB6) son los que determinan el número de neurona. Como todas las tarjetas base de las neuronas son iguales, la dirección queda fijada mediante un DIP-Switch ubicado en la misma, de esta manera la tarjeta base se puede programar para funcionar como cualquier neurona. Bajo este esquema se pueden direccionar 16 neuronas, aunque el sistema sólo cuenta con diez. Finalmente, la última línea del puerto B (RB7) se utiliza para duplicar el mapa de puertos es decir, en lugar de sólo tener 16 posibles neuronas, existe la posibilidad de direccionar hasta 32. Para ello bastará con cambiar la programación de la GAL ubicada en las otras 16 tarjetas para responder a una señal SEL7 pero invertida.

RB2	RB1	RB0	RA1	RA0	PUERTO
0	0	0	0	0	DAC SUMADOR-A W1
0	0	0	0	1	DAC SUMADOR-A W2
0	0	0	1	0	DAC SUMADOR-A W3
0	0	0	1	1	DAC SUMADOR-A W4
0	0	1	0	0	DAC SUMADOR-B W5
0	0	1	0	1	DAC SUMADOR-B W6
0	0	1	1	0	DAC SUMADOR-B W7
0	0	1	1	1	DAC SUMADOR-B W8
0	1	0	0	0	DAC INTEGRADOR TAU
0	1	0	0	1	DAC INTEGRADOR POT. REPOSO
0	1	0	1	0	DAC INTEGRADOR OFFSET
0	1	0	1	1	DAC INTEGRADOR NO UTILIZADO
0	1	1	0	0	DAC F. DE ACTIV-A RAMPA SATURACION
0	1	1	0	1	DAC F. DE ACTIV-A RAMPA OFFSET
0	1	1	1	0	DAC F. DE ACTIV-A RAMPA PENDIENTE
0	1	1	1	1	DAC F. DE ACTIV-A ESCALON UMBRAL
1	0	0	0	0	DAC F. DE ACTIV-B TANH SATURACION
1	0	0	0	1	DAC F. DE ACTIV-B TANH OFFSET
1	0	0	1	0	DAC F. DE ACTIV-B TANH PENDIENTE
1	0	0	1	1	DAC F. DE ACTIV-B NO UTILIZADO
1	0	1	0	0	DAC GEN. PULSOS RETARDO
1	0	1	0	1	DAC GEN. PULSOS DURACION
1	0	1	1	0	DAC GEN. PULSOS UMBRAL
1	0	1	1	1	DAC GEN. PULSOS NO UTILIZADO
1	1	0	0	0	LATCH1 INTEGRADOR SW ANALOGICO+LEDS
1	1	0	0	1	LATCH1 INTEGRADOR SW ANALOGICO+LEDS
1	1	0	1	0	LATCH1 INTEGRADOR SW ANALOGICO+LEDS
1	1	0	1	1	LATCH1 INTEGRADOR SW ANALOGICO+LEDS
1	1	1	0	0	LATCH2 F. ACTIV. SW ANALOGICO+LEDS
1	1	1	0	1	LATCH2 F. ACTIV. SW ANALOGICO+LEDS
1	1	1	1	0	LATCH2 F. ACTIV. SW ANALOGICO+LEDS
1	1	1	1	1	LATCH2 F. ACTIV. SW ANALOGICO+LEDS

Tabla 3. Mapa de puertos.

Con esto se ha descrito las funciones de la tarjeta de interfaz y el diseño de las tarjetas base de cada neurona. En el siguiente punto se discutirá el diseño del gabinete.

II . 5 DISEÑO DEL GABINETE

Una vez que se han desarrollado las diferentes etapas que constituyen el modelo de neurona y se ha diseñado la interfaz de control y comunicación con la computadora personal. se procede a elaborar el esquema que permita el ensamble de un sistema formado por 10 neuronas. Utilizando como punto de partida los diferentes bloques que constituyen una neurona, se plantea el desarrollo de un sistema con arquitectura modular basada en tarjetas del tipo eurocard, de tal manera que se puedan insertar o desconectar fácilmente simplificando así las labores de calibración, mantenimiento y actualización. La arquitectura modular permite alterar parcialmente el modelo de neurona sin tener que rediseñar toda la computadora y permite además realizar sólo modificaciones parciales o temporales al esquema general del proyecto.

El sistema final consiste de 10 módulos (neuronas) cada uno de los cuales esta subdividido en 4 bloques. Cada módulo cuenta con una tarjeta base donde se encuentra la lógica de descodificación y los reforzadores de bus (buffers). Cada uno de los bloques del modelo (sumador, integrador, funciones de activación y generador de pulso) está implantado en una tarjeta impresa tipo eurocard que puede montarse o desmontarse frontalmente.

Los bloques del módulo se acoplan a una tarjeta base como la ilustrada en la figura 45. Se pueden observar los cuatro slots donde se insertarán igual número de bloques de los que forma parte la neurona, así como el espacio para la lógica de descodificación y un DIP-switch que sirve para fijar la dirección base del módulo. Los slots del integrador y de las funciones de activación se encuentran a la misma altura debido a que son perfectamente intercambiables; es decir, el diseño tiene contemplado que las posiciones del bloque de integración y el de funciones de activación se puedan invertir para dar con ello flexibilidad al modelo de neurona. De esta manera se puede construir una neurona cuyos bloques entrada salida sean sumador, función de activación, integrador y generador de pulso. Por otro lado, los slots del bloque sumador y el bloque generador de pulso se encuentran ubicados a diferente nivel para impedir una inserción errónea.

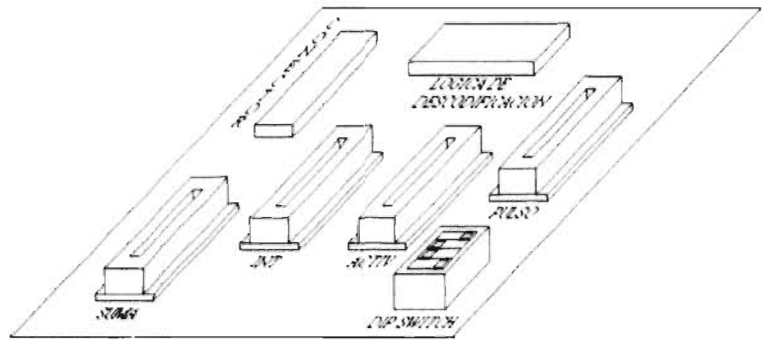


Fig. 45. Tarjeta base del módulo.

En la fig. 46 se muestra el diagrama de interconexión de las etapas que constituyen una neurona y de las conexiones principales a las que se tiene acceso desde el tablero.

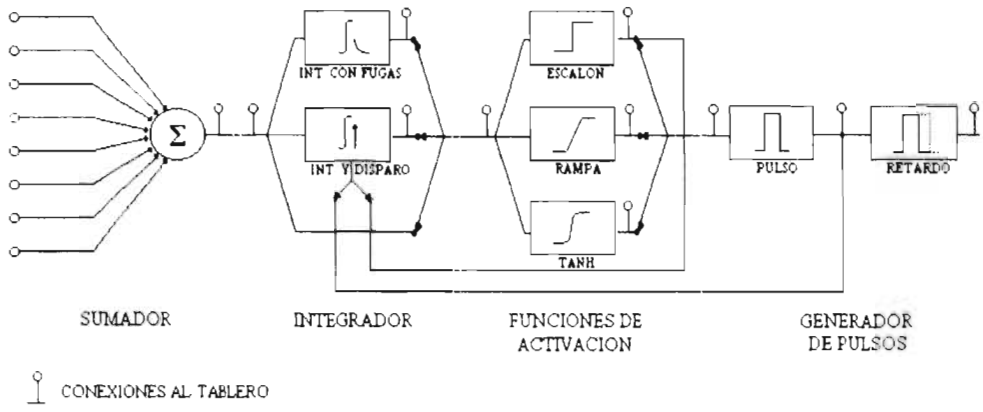


Fig. 46. Diagrama de interconexión de las etapas.

Como se mencionó anteriormente, cada módulo está formado por los cuatro bloques que constituyen el modelo de neurona. En la figura 47 se puede observar un esquema de la carátula de cada módulo donde se indica además las ubicaciones de los conectores de entrada y salida, así como los LEDs que indican el modo de operación. Se muestran también los interruptores para seleccionar una fuente externa de control para la constante de tiempo del integrador y el nivel de umbral de la función de activación escalón. En la figura 48 se muestra la función de cada una de las conexiones de las que consiste cada módulo, así como la función de los LEDs.

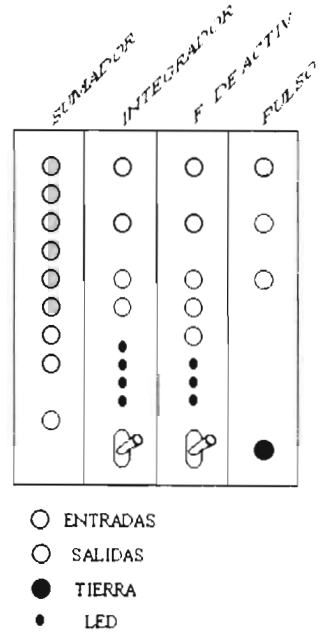


Fig. 47. Carátula del módulo.

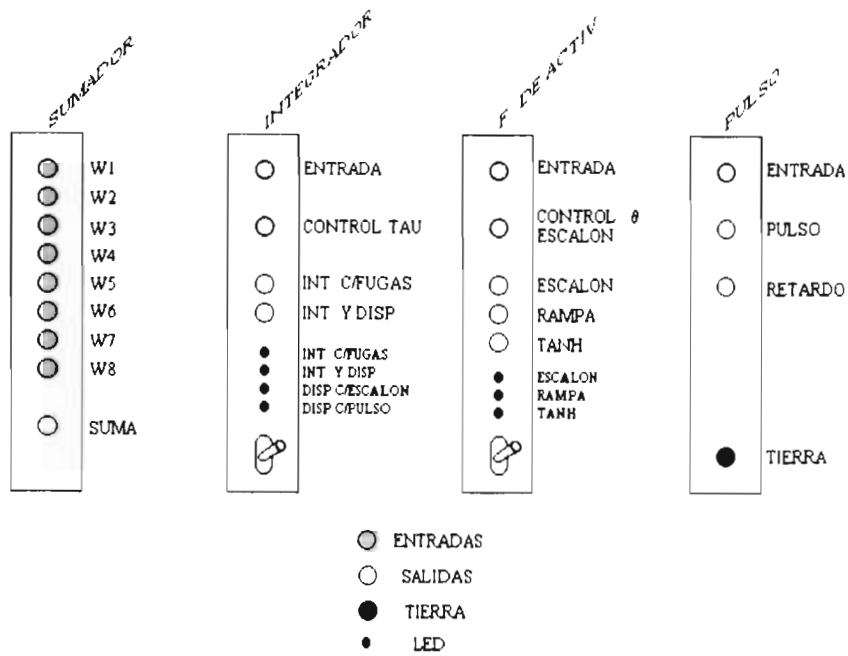


Fig. 48. Detalle de las conexiones del módulo.

En la figura 49 se muestra un diagrama del chasis de la computadora analógica donde se observan los diez módulos de que está constituida. En ese chasis se encuentran colocados además, el interruptor de restablecimiento (reset) de la interfaz, el interruptor de encendido

y unos LEDs que indicarán el estado de la comunicación entre la computadora analógica y la computadora personal. Por la parte de atrás se encuentran el conector serial y la entrada para la alimentación.

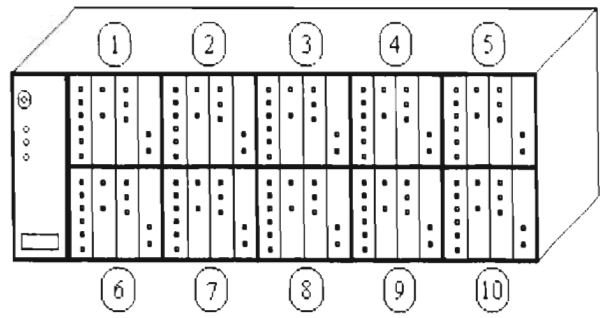


Fig. 49. Diagrama del chasis de la computadora analógica.

Finalmente, en la fig. 50 se muestra una fotografía de la tarjeta de interfaz y un módulo (neurona) ensamblados en circuito impreso. Se observa el conector DB9 para comunicación serial, un cable plano que conecta la interfaz de comunicación con la tarjeta base y las cuatro tarjetas de las cuatro etapas que comprende la neurona (sumador, integrador, funciones de activación y generador de pulsos).

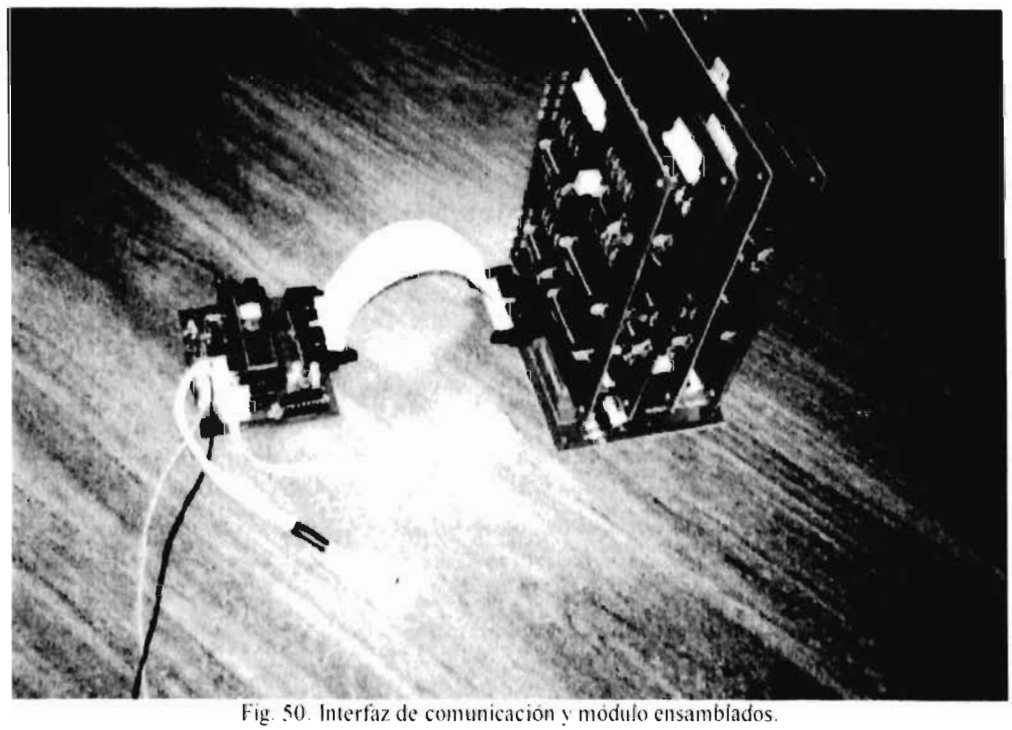


Fig. 50. Interfaz de comunicación y módulo ensamblados.

En los apéndices que se incluyen al final de este trabajo se encuentran los diagramas esquemáticos completos de los circuitos diseñados, los diagramas de circuitos impresos y la lista de partes.

II . 6 BIBLIOGRAFIA

[1] C. F. Stevens. "The neuron". *Scientific American.* 1979.

[2] S.Soclof. "Design and applications of analog integrated circuits". Prentice Hall. Englewood Cliffs. New Jersey. 1991.

[3] H. Yanai and Y. Sawada. "Integrator neurons for analog neural networks". *IEEE Trans. Circuits Syst.*, vol. 37, no. 6, pp. 854-856. Jun. 1990.

[4] A. Hamilton, A. F. Murray, H. M. Reekie, and L. Tarassenko. "Working analog pulse-firing neural network chips". In *VLSI for artificial intelligence and neural networks.* Edited by J. G. Delgado-Frias and W. R. Moore, Plenum Press, New York. pp. 225-234. 1991.

[5] A. Hamilton, A. F. Murray, and H. M Reekie. "Pulse-firing VLSI neural circuits for fast image pattern recognition". In *VLSI for artificial intelligence and neural networks.* Edited by J. G. Delgado-Frias and W. R. Moore, Plenum Press, New York. pp. 235-244. 1991.

[6] D. Patranabis, C. Bakshi and S. Ghosh. "True integrators with DVCCS". *IEEE. Trans. Instrum. Meas.*, vol. IM-35, no. 3. pp. 245-248. Sept. 1986.

[7] E. Sánchez-Sinencio, R. L. Geiger, and H. Nevarez-Lozano. "Generation of continous-time two integrator loop OTA filter structures". *IEEE Trans. Circuits Syst.*, vol. CAS-35, pp. 936-945, Aug. 1988.

[8] RCA."Integrated circuits for linear applications". RCA Corp. 1986.

[9] Maxim. "New releases databook vol. II". Maxim Inc. 1993.

[10] Analog Devices."Linear products databook". Analog Devices Inc. 1990.

[11] Analog Devices. "Data conversion products databook". Analog Devices Inc. 1989/1990.

[12] National Semiconductor. "Operational amplifiers databook". National Semiconductor Corp. 1993.

[13] M. A. Bañuelos S., and J. L. Pérez S. "Analog computer based on neuron electronic models". In memories of *Sian ka'an International Conference. The first joint Mexico-US International Workshop on Neural Networks and Neurocontrol.* Playa del Carmen, Quintana Roo, Mexico. Sept. 5-15, 1995. pp. 64-76.

[14] M. A. Bañuelos S., and J. L. Pérez S. "Analog computer for neural networks". In memories of *Sian ka'an 97 International Conference. The second joint Mexico-US International Workshop on Neural Networks and Neurocontrol.* Playa del Carmen, Quintana Roo, Mexico. Aug. 19-29, 1997. pp. 28-38.

[15] Microchip. *PIC16C7X 8-bit CMOS microcontrollers with A/D converter datasheet.* DS30390D. Microchip Technology Inc. 1996.

III PROGRAMA DE CONTROL

En el capítulo anterior se describió el desarrollo de la electrónica asociada al sistema de cómputo analógico para redes neuronales. Cada neurona tiene 20 parámetros ajustables y tres opciones de operación. La función del programa de control es proporcionar al usuario del sistema una interfaz amigable y fácil de utilizar para poder controlar los 200 parámetros del mismo. Para realizar el control de estas funciones se elaboró un programa bajo ambiente Windows, debido a que este tipo de ambiente presenta un modo de operación intuitivo y estándar para el usuario. El lenguaje de programación que se utilizó fue Visual Basic (versión 4.0) debido a la facilidad que proporciona para elaborar programas de ambiente gráfico. El programa fue desarrollado para operar bajo el sistema operativo Windows 95, para hacerlo compatible con este sistema que tiene un uso muy extendido.

III . 1 ESTRUCTURA DEL PROGRAMA

La programación en Visual Basic sigue una metodología denominada programación orientada a eventos: esto es, en una interfaz gráfica de usuario (GUI, por sus siglas en inglés) se encuentran uno o varios objetos con los cuales interactúa el usuario para determinar el flujo del programa. Bajo esta filosofía, un programa se desarrolla conectando código a un evento significativo dentro de un objeto; por ejemplo, hacer clic sobre un botón.

La estructura del programa la podemos dividir entonces en una ventana principal junto con los objetos y eventos asociados a ella y en ventanas secundarias a las cuales se invoca mediante eventos dentro de la ventana principal. Las funciones básicas de la ventana principal son: el manejo de archivos, la selección de la neurona, la selección de la etapa a modificar (sumador, integrador, función de activación o generador de pulsos), y el despliegue de todos los parámetros (ver fig. 1).

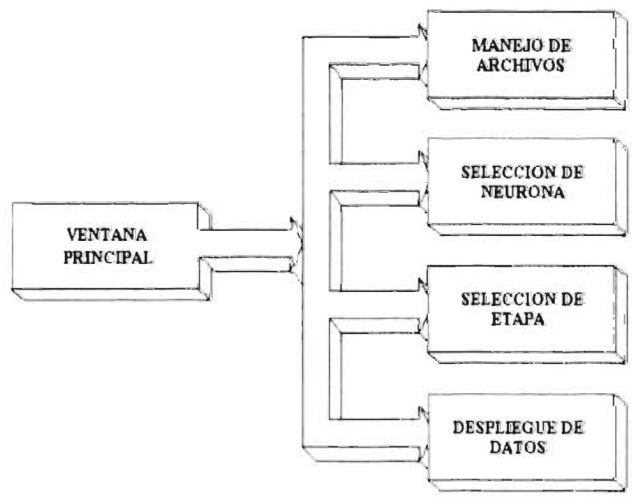
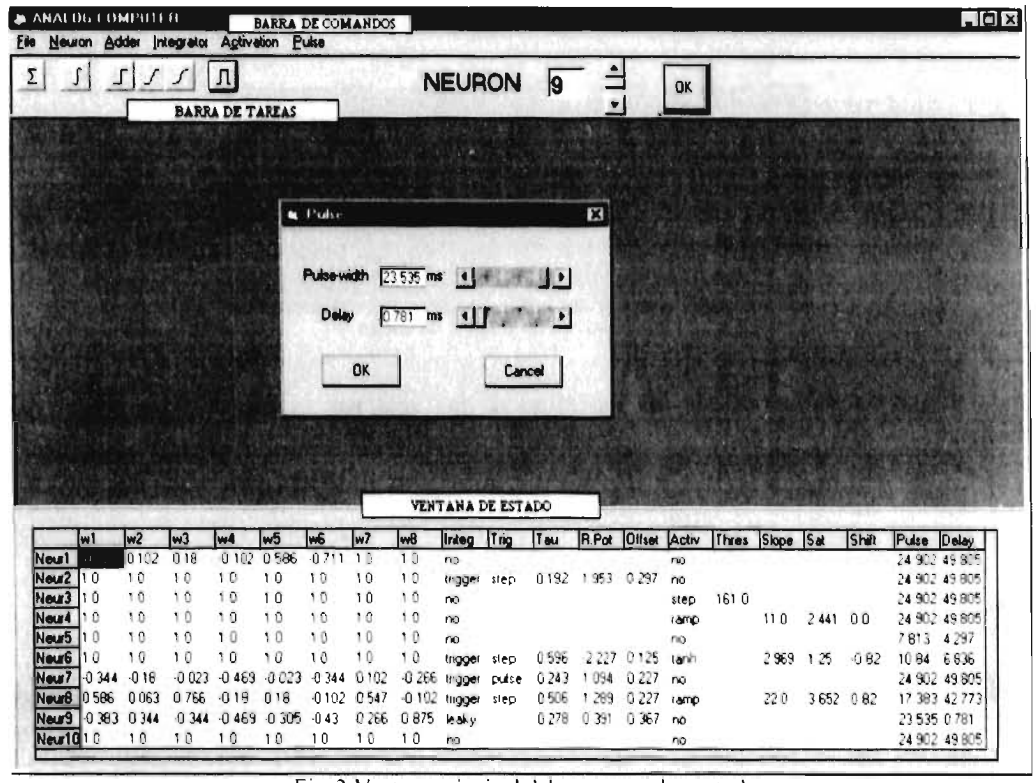


Fig. 1. Funciones de la ventana principal.

En la ventana principal se encuentra una barra de comandos, una barra de tareas y una ventana de estado donde se despliegan los valores de los parámetros y opciones seleccionadas de la computadora (ver fig. 2). En la barra de comandos se encuentran los menús de manejo de archivos, selección de neurona, ajuste de parámetros del sumador, de las funciones de activación y del generador de pulsos. En la barra de tareas se encuentran unos botones que invocan a las ventanas de ajuste de parámetros de las diferentes etapas de la neurona (sumador, integrador, funciones de activación y generador de pulsos). Así mismo, se encuentra una barra de deslizamiento que permite seleccionar el número de neurona y también un botón que es sirve para transferir la información a la computadora analógica. De esta manera, los parámetros se actualizarán hasta el momento en que sea presionado el botón "OK"; esto permite sincronizar la actualización de varios parámetros o bien corregirlos si se detecta algún error. En la ventana de estado se puede visualizar el valor de todos los parámetros ajustables y el valor de las opciones seleccionadas, de tal manera que en todo momento se muestra la configuración de la computadora analógica. Debido a la cantidad de datos que se despliegan, se recomienda utilizar un despliegue de video de 1024 x 748 pixeles, ya que de lo contrario no se podrán visualizar todos los datos simultáneamente.



	w1	w2	w3	w4	w5	w6	w7	w8	Integ	Trig	Tau	R.Pot	Offset	Activ	Thres	Slope	Sat	Shift	Pulse	Delay
Neur1		0.102	0.18	-0.102	0.586	-0.711	1.0	1.0	no					no					24.902	49.805
Neur2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	trigger	step	0.192	1.953	0.297	no					24.902	49.805
Neur3	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					step	161.0				24.902	49.805
Neur4	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					ramp		11.0	2.441	0.0	24.902	49.805
Neur5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					no					7.813	4.297
Neur6	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	trigger	step	0.596	2.227	0.125	tanh		2.969	1.25	-0.82	10.84	6.836
Neur7	-0.344	-0.18	-0.023	-0.469	-0.023	-0.344	0.102	-0.266	trigger	pulse	0.243	1.094	0.227	no					24.902	49.805
Neur8	0.586	0.063	0.766	-0.18	0.18	-0.102	0.547	-0.102	trigger	step	0.506	1.289	0.227	ramp		22.0	3.652	0.82	17.383	42.773
Neur9	-0.383	0.344	-0.344	-0.469	-0.305	-0.43	0.266	0.875	leaky		0.278	0.391	0.367	no					23.535	0.781
Neur10	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					no					24.902	49.805

Fig. 2 Ventana principal del programa de control.

Mediante la ventana principal se activan las ventanas secundarias que entre otras funciones tendrán la de modificar los parámetros de la neurona seleccionada, transmitir los datos por el puerto serial hacia la interfaz electrónica y realizar el manejo de archivos de configuración.

III . 2 MODULOS DEL PROGRAMA

El programa cuenta con cuatro ventanas secundarias relativas al ajuste de los parámetros de la computadora analógica: etapa sumadora, integrador, funciones de activación y generador de pulsos. Al accesar estas ventanas se podrán variar los parámetros correspondientes a esa etapa del modelo, para la neurona seleccionada en la barra de tareas del menú principal.

Las ventanas secundarias se pueden habilitar mediante la barra de comandos o mediante los botones de la barra de tareas. En cada uno de los botones se encuentra un ícono que indica la etapa asociada. Además, al posicionar el cursor sobre el botón aparece un letrero con el nombre de la etapa (ver fig. 3).

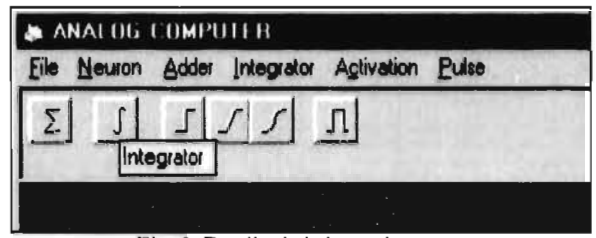


Fig. 3. Detalle de la barra de tareas.

A continuación se describirán las funciones de las ventanas asociadas a las etapas del modelo.

III . 2 . 1 ETAPA SUMADORA

La primera etapa del modelo la constituye el sumador. En esta ventana es posible modificar los valores de los pesos de las ocho entradas entre -0.992 y 1.000 divididos en 256 pasos (ver fig. 4), por lo que los incrementos serán de 0.008 unidades aproximadamente. Al posicionar el cursor sobre las flechas y presionar el botón izquierdo del ratón, se consiguen incrementos equivalentes a un paso: mientras que si se coloca sobre la barra de deslizamiento éstos serán de 10 pasos.

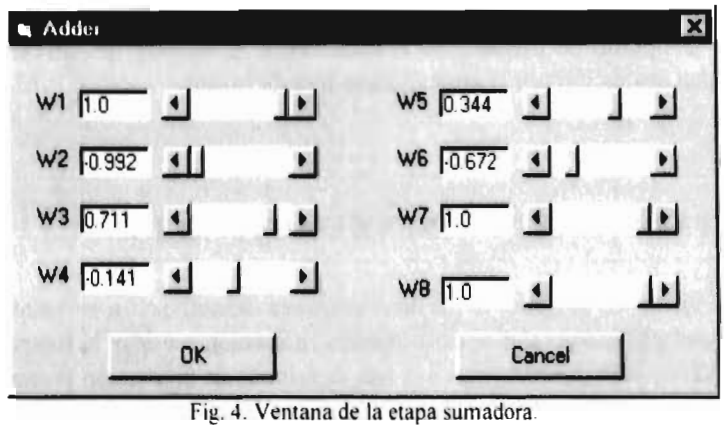


Fig. 4. Ventana de la etapa sumadora.

Al presionar el botón "OK" los datos reemplazarán al contenido anterior del buffer del programa, mientras que si se presiona el botón "Cancel" se conservarán los valores anteriores. Sin embargo los datos no serán transferidos a la computadora analógica sino hasta que se presione el botón "OK" ubicado en la barra de tareas.

III . 2 . 2 ETAPA INTEGRADORA

El siguiente módulo es la etapa integradora. En esta etapa es posible modificar la constante de tiempo, el potencial de reposo y ajustar un voltaje de offset que sirve para calibrar el nivel de DC de la salida del integrador. La constante de tiempo se puede variar entre 149 μs y 22.4 ms, el potencial de reposo entre -5.0 y 4.961 V y el offset entre -0.992 y 1.0 V en 256 pasos para todos los casos. Además se puede seleccionar entre integrador con disparo, integrador con fugas, o bien deshabilitar la función integrador. Esto último se logra al puentear la etapa mediante los interruptores analógicos incluidos en la tarjeta correspondiente, de manera tal que se puede deshabilitar el integrador desde el programa sin necesidad de modificar las conexiones presentes.

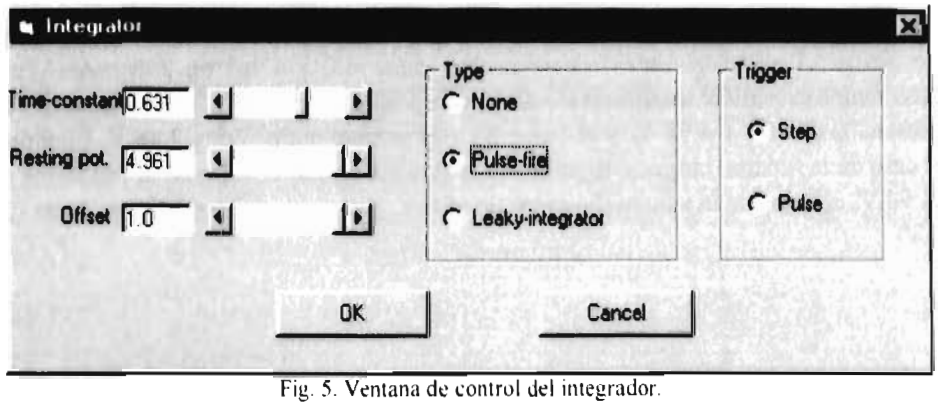


Fig. 5. Ventana de control del integrador.

En el caso de que se seleccione el integrador con disparo, se habilita un marco donde se puede escoger la opción de disparo, ya sea mediante la función de activación escalón o mediante el pulso producido por la etapa generadora de pulsos.

III . 2 . 3 ETAPA DE FUNCIONES DE ACTIVACION

La ventana de ajuste de parámetros de las funciones de activación se puede invocar con cualquiera de los tres botones que corresponden a la función escalón, la rampa y la tangente hiperbólica. Esto se hizo debido a que las tres funciones de activación corresponden a una sola etapa del modelo y físicamente están implementadas en la misma tarjeta. En la ventana se pueden observar varios marcos (ver fig. 6). En uno de ellos se puede seleccionar el tipo de función de activación que será usada o bien si se desea puentear este módulo.

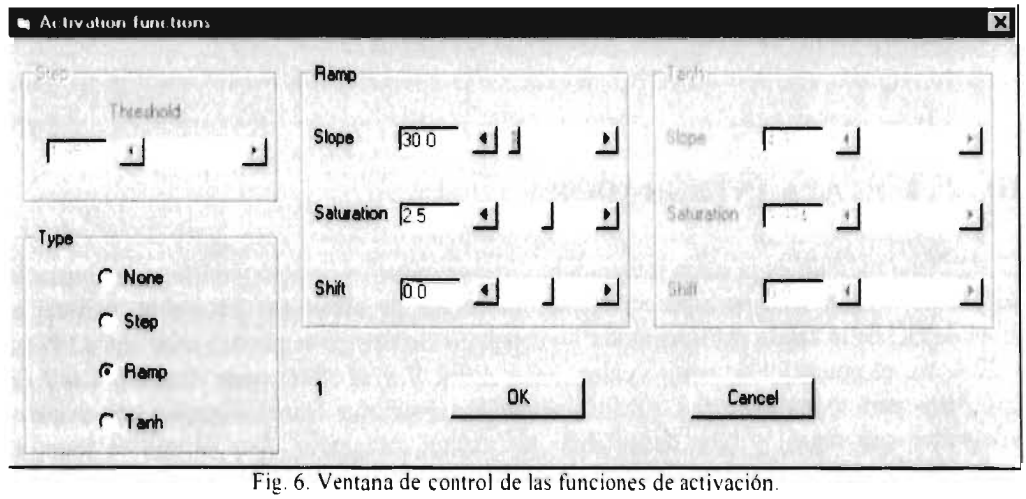


Fig. 6. Ventana de control de las funciones de activación.

En el momento que se selecciona una función, se activa el marco correspondiente y con ello se habilita el ajuste de los parámetros correspondientes. En el caso de la función escalón se puede ajustar el nivel de umbral de la misma entre -5.0 y 4.961 en 256 pasos. Para la función rampa es posible modificar el valor de la pendiente entre 0 y 255, el valor del nivel de saturación entre 0 y 4.98 V, y el valor del corrimiento entre -2.5 y 2.48 V. Finalmente, en el caso de la función tangente hiperbólica es posible ajustar el valor de la pendiente entre 0 y 4.98 V, el valor de la saturación entre 0 y 4.0 V, y el valor del corrimiento entre -2.5 y 2.48 V.

III . 2 . 4 ETAPA GENERADORA DE PULSOS

La última etapa del modelo es la etapa generadora de pulsos, cuya ventana de control se muestra en la figura 7. En esta ventana es posible modificar el ancho del pulso (entre 0 y 24.9 ms), y la duración del retardo (entre 0 y 49.8 ms).

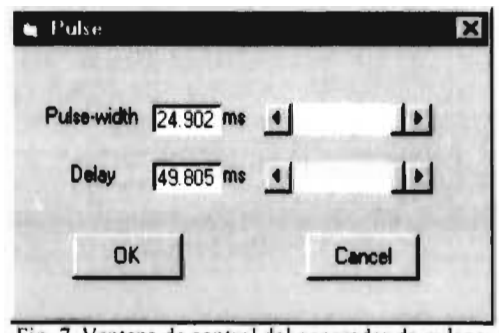
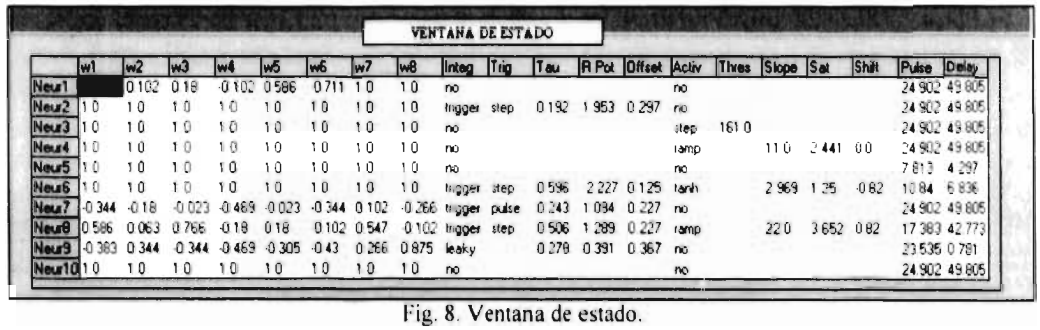


Fig. 7. Ventana de control del generador de pulsos.

III . 2 . 5 VENTANA DE ESTADO

En la figura 8 se muestra la ventana de estado, misma que sirve para desplegar los valores de los parámetros de ajuste y opciones seleccionadas para la neurocomputadora. En esta ventana se cuenta con un renglón por cada una de las 10 neuronas del sistema. Las columnas w1 a w8 indican el valor de los pesos de la etapa sumadora. La siguiente columna, "integ", muestra la configuración de la etapa integradora, desplegando la palabra "no" en el caso de que el integrador se encuentre desactivado (puenteado), la palabra "trigger" en el caso de un integrador y disparo, y la palabra "leaky" para un integrador con fugas. Si se ha seleccionado un integrador y disparo, en la siguiente columna "Trig" aparecerá la etiqueta "step" para el caso de que la señal de disparo se tome de la función escalón, la etiqueta "pulse" aparecerá si se toma la señal de disparo del circuito generador de pulsos, o bien, si no se trata de un integrador y disparo, el recuadro permanecerá vacío. Las columnas 10, 11 y 12 despliegan la constante de tiempo ("Tau"), el potencial de reposo ("R. Pot") y el ajuste de offset ("Offset") del integrador, respectivamente. En el caso de que no se encuentre activado el integrador, estas tres columnas permanecerán vacías. La siguiente columna, "Activ", muestra el tipo de función de activación que se ha seleccionado. La etiqueta "step" indica una función escalón, "ramp" una función rampa-saturación, "tanh" una función tangente hiperbólica y "no" que el módulo se encuentra desactivado y la entrada será conectada directamente a la salida. La columna denominada "Thres" mostrará el nivel de umbral en el caso de que se haya seleccionado una función de activación tipo escalón, de otra manera permanecerá vacía. Cuando se haya seleccionado la función de activación rampa o tangente hiperbólica, las columnas denominadas "Slope",

"Sat" y "Shift", desplegarán el valor correspondiente a la pendiente, el nivel de saturación y el corrimiento, respectivamente. Si no se seleccionó ninguna de estas dos funciones, el recuadro permanecerá vacío. Las últimas dos columnas se denominan "Pulse" y "Delay" y su contenido representa el valor del ancho del pulso y de la duración del retardo respectivamente.


VENTANA DE ESTADO

	w1	w2	w3	w4	w5	w6	w7	w8	Integ	Trig	Tau	R Pot	Offset	Activ	Thres	Slope	Sat	Shift	Pulse	Delay
Neur1		0.102	0.18	-0.102	0.586	0.711	1.0	1.0	no					no					24.902	49.805
Neur2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	trigger	step	0.192	1.953	0.297	no					24.902	49.805
Neur3	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					step	161.0				24.902	49.805
Neur4	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					ramp		11.0	2.441	0.0	24.902	49.805
Neur5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					no					7.813	4.297
Neur6	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	trigger	step	0.586	2.227	0.125	tanh		2.969	1.25	-0.82	10.84	6.836
Neur7	-0.344	-0.18	-0.023	-0.469	-0.023	-0.344	0.102	-0.266	trigger	pulse	0.243	1.094	0.227	no					24.902	49.805
Neur8	0.586	0.063	0.766	-0.18	0.18	0.102	0.547	0.102	trigger	step	0.506	1.289	0.227	ramp		22.0	3.652	0.82	17.383	42.773
Neur9	-0.383	0.344	-0.344	-0.469	-0.305	-0.43	0.266	0.875	leaky		0.278	0.391	0.367	no					23.535	0.781
Neur10	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	no					no					24.902	49.805

Fig. 8. Ventana de estado.

III . 3 ARCHIVOS DE CONFIGURACION

Una de las ventajas que proporciona el control de parámetros desde una computadora es la posibilidad de generar archivos de datos donde se almacenen los valores de todos los parámetros. A estos archivos los hemos denominado archivos de configuración. Para guardar o leer un archivo de configuración basta con accesar la opción desde el menú de comandos en la parte superior de la ventana principal.

Como se mencionó anteriormente, el sistema es capaz de modificar 20 parámetros y 3 opciones de conexión por cada neurona; esto hace un total de 200 parámetros y 30 opciones que se guardarán en un archivo de 230 elementos. En la tabla 1 se muestran las variables asociadas con los parámetros.

```
Global peso(10, 8) As Byte 'peso(neur, num de entrada)
Global inttip(10) As Byte 'tipo de integrador
Global disparo(10) As Byte 'tipo de disparo del integrador
Global tau(10) As Byte 'Constante de tiempo del integrador
Global pot(10) As Byte 'Potencial de reposo del integrador
Global offset(10) As Byte 'ajuste de offset del integrador
Global acttip(10) As Byte 'tipo de función de activación
Global umbral(10) As Byte 'Umbral del escalón
Global pend(10) As Byte 'Pendiente de la rampa
Global sat(10) As Byte 'Saturación de la rampa
Global corr(10) As Byte 'Corrimiento de la rampa
Global pends(10) As Byte 'Pendiente de la sigmoide
Global sats(10) As Byte 'Saturación de la sigmoide
Global corrs(10) As Byte 'Corrimiento de la sigmoide
Global pulso(10) As Byte 'Duración del pulso
Global retardo(10) As Byte 'Duración del retardo
```

Tabla 1. Variables de configuración.

El contenido de las variables se almacena en archivos de texto con extensión .DAT los cuales se pueden cargar o guardar utilizando el menú "file" de la barra de comandos. En la fig. 9 se muestra un ejemplo de la ventana de manejo de archivos.

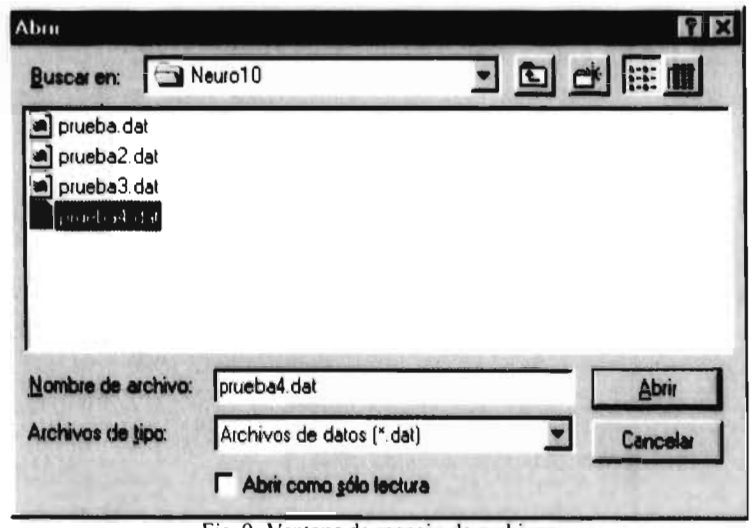


Fig. 9. Ventana de manejo de archivos.

De esta manera, se utilizan ventanas típicas de una aplicación bajo Windows para el manejo de los archivos de configuración, lo cual hace más fácil el manejo del programa.

Hasta aquí se ha dado una breve descripción del funcionamiento del programa de control de la neurocomputadora. En el apéndice X se enlistan las principales rutinas del programa desarrollado.

III . 4 BIBLIOGRAFIA

[1] Microsoft. *Microsoft Visual Basic version 4.0 Programmer's guide.* Microsoft Corp.1995.
[2] Microsoft. *Microsoft Visual Basic version 4.0 Language reference.* Microsoft Corp.1995.
[3] Microsoft. *Microsoft Visual Basic version 4.0 Professional features.* Microsoft Corp.1995.
[4] Heyman M. *La esencia de Visual Basic 4.* Prentice Hall Hispanoamericana. 1996.

RESULTADOS
Y
CONCLUSIONES

Se ha desarrollado una neurocomputadora analógica con parámetros controlados digitalmente. La misma consiste de 10 neuronas divididas es cuatro etapas cada una: sumador, integrador, función de activación no-lineal y generador de pulsos. Cuenta con un total de 200 parámetros ajustables digitalmente por medio de convertidores digital-analógicos y 30 opciones de configuración, los cuales son controlados desde una computadora personal con un programa que corre bajo la plataforma Windows 95. Esto, aunado con la característica de poder almacenar el conjunto de parámetros en archivos de computadora, el sistema proporciona una herramienta ágil para el estudio de redes neuronales dinámicas en tiempo real.

La etapa sumadora diseñada tiene ocho entradas con ganancias controladas individualmente ajustables entre -0.992 y 1.0. Puede procesar señales cuadradas de hasta 1 kHz con mínima distorsión y presenta un error de offset a la salida de 7 mV.

Se diseñó una etapa integradora basada en un amplificador operacional de transconductancia, la cual tiene la ventaja de permitir un control continuo de la constante de tiempo, por lo cual el circuito diseñado puede ser utilizado junto con un convertidor digital analógico de cualquier resolución, o bien, directamente mediante la aplicación de una señal continua, tal y como se contempla en el tablero de conexiones. La constante de tiempo puede variarse en una razón de 1:200 y funcionó satisfactoriamente con señales de hasta 10 kHz. El circuito implantado permite además el control del potencial de reposo y cancelación de errores de offset vía programación. El integrador puede ser configurado como integrador con fugas o integrador y disparo, teniendo en este último caso la posibilidad de utilizar como señal de control del disparo a la proveniente de la función de activación escalón o la de del circuito generador de pulsos.

El circuito función de activación escalón permite ajustar el umbral de activación entre -5.0 y 4.96 V y funciona con mínima distorsión con señales de hasta 10 kHz. Adicionalmente, por medio del tablero de conexiones es posible controlar el nivel de umbral con una señal analógica externa, con lo cual se pueden implantar umbrales dependientes del tiempo.

El circuito función de activación rampa-saturación permite ajustar la pendiente en un intervalo de 1 a 255, el nivel de saturación en un intervalo de 0 a 5 V y el corrimiento de la función entre -2.5 V y +2.5 V. La implementación se basa en circuitos recortadores que evitan que la salida de los amplificadores operacionales utilizados se sature y aumente el tiempo de respuesta. De esta manera se logra una frecuencia máxima de operación de 1 kHz.

Para la implantación de la función de activación tangente hiperbólica se utilizó un amplificador operacional de transconductancia, debido a que esta su relación entrada-salida corresponde a una tangente hiperbólica. Los parámetros que se pueden ajustar son la pendiente (de 0 a 5), el nivel de saturación (de 0 a 4 V) y el corrimiento (de -2.5 a +2.5 V). El circuito procesa adecuadamente señales de hasta 1 kHz dado que se presenta un desfasamiento menor a 1 %.

La etapa de generación de pulsos diseñada permite generar pulsos con ancho ajustable entre 0.1 y 25 ms, o bien, pulsos de iguales características pero retardados en el tiempo de entre 0.2 y 50 ms. El diseño, que se basa en temporizadores del tipo LM555, tiene la ventaja de que el control de los parámetros es lineal.

Se desarrolló también una interfaz de comunicación serial basada en el microcontrolador tipo RISC PIC16C74 y con el cual se recibe la información de ajuste de parámetros desde la computadora personal y se envía a los DACs y Latches de la neurocomputadora. Para minimizar el número de componenetes, la lógica de decodificación se implementó utilizando un GAL.

Se diseñó un sistema modular utilizando tarjetas con conectores tipo Eurocard, de tal manera que se facilite la calibración y mantenimiento del sistema. El modelo de neurona está dividido en cuatro etapas y cada una de ellas fue implantada en una tarjeta independiente que se conecta a una tarjeta base.

El total de los parámetros se controla mediante el uso de DACs, y para minimizar el número de componentes se utilizó el C.I MAX505 que es un convertidor cuádruple de 8 bits.

Como se puede ver en este trabajo, la generación de funciones matemáticas mediante circuitos analógicos puede llevar a soluciones complejas cuando se quieren controlar numerosos parámetros y minimizar diversos errores como lo podrían ser los errores de offset. Es por ello que una etapa posterior al proyecto podría ser la implantación del mismo en VLSI, dando además la posibilidad de construir sistemas con un mayor número de neuronas.

El programa de control desarrollado en Visual Basic 4.0 proporciona una interfaz ágil para el ajuste, visualización y almacenamiento de los parámetros.

De los resultados obtenidos se observa que una de las limitaciones importantes es el ancho de banda de algunas etapas del sistema. Por ejemplo, en el caso de la etapa de funciones activación, el objetivo es implantar electrónicamente una relación entrada-salida instantánea, y dado que los dispositivos electrónicos no responden instantáneamente, incluso a frecuencias del orden de algunos cientos de Hertz ya se percibian errores debido al desfasamiento entre la entrada y la salida del circuito. Esto, podría pensarse como una clara indicación de que se requerían circuitos de mayor velocidad; sin embargo, ésto no es necesariamente cierto dado que los sistemas neuronales biológicos funcionan en un ancho de banda de algunos cientos de Hertz.

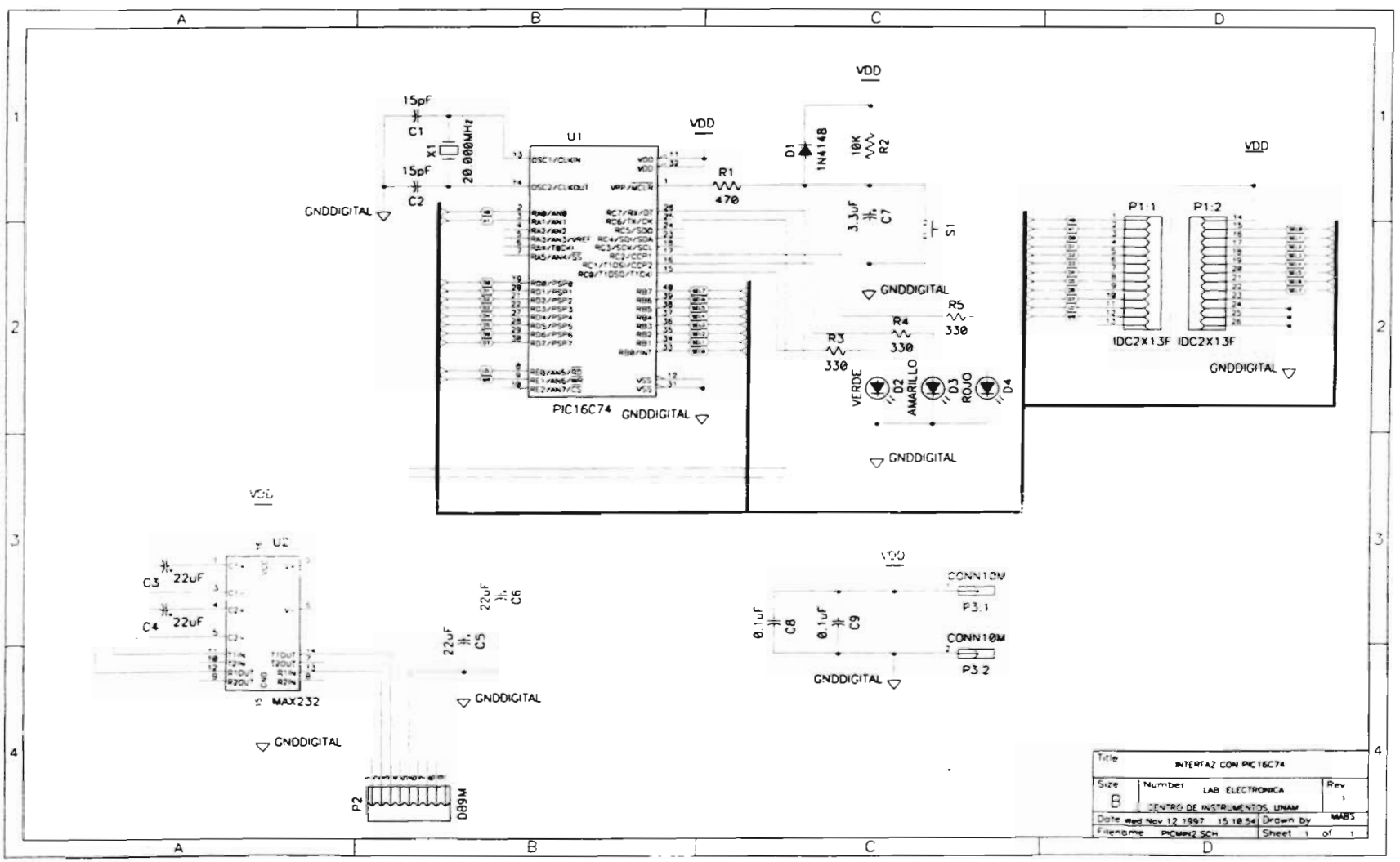
Las diferencias existentes entre una función matemática y su implantación electrónica abren el campo a la investigación de nuevos modelos matemáticos de circuitos electrónicos. Se deberán adecuar los modelos matemáticos a la implantación electrónica y no a la inversa. Esto permitirá generar nuevos modelos de neuronas y nuevas arquitecturas de redes, que al

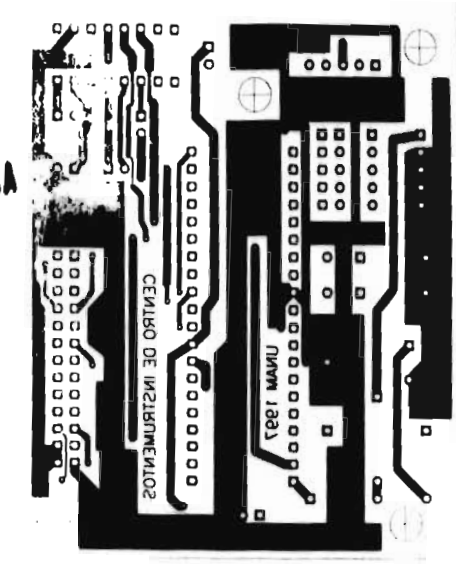
tener en cuenta la implantación electrónica, faciliten el desarrollo de ésta y sean capaces de presentar alternativas a muchos de los problemas aún no resueltos en el área.

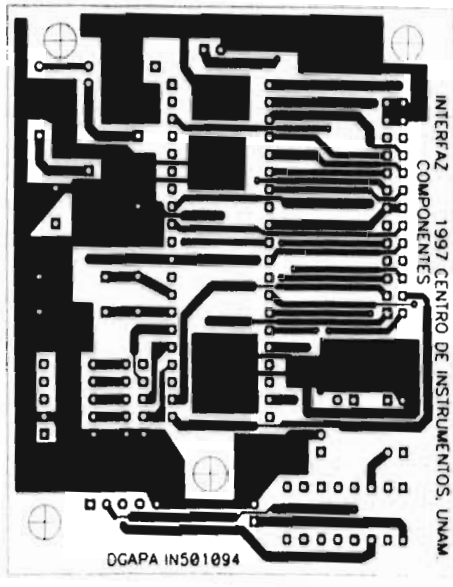
APENDICE I

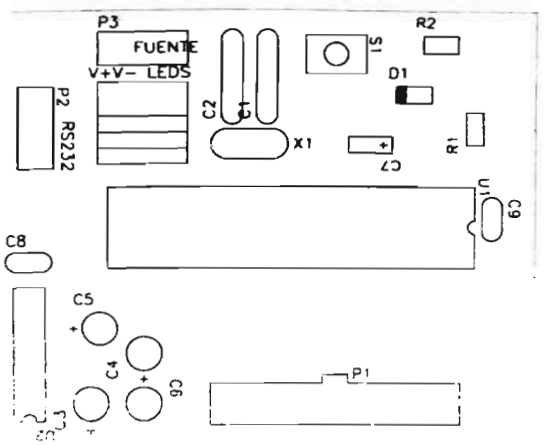
DIAGRAMAS ESQUEMATICOS
Y
CIRCUITOS IMPRESOS

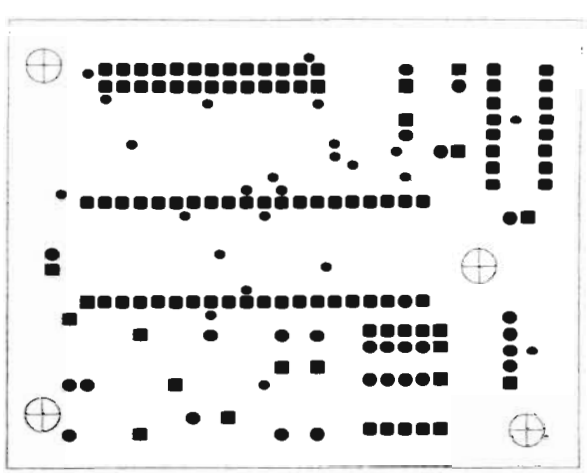
TARJETA DE INTERFAZ


A
B
C
D
15pF
C1
X1
20.000MHz
15pF
C2
GNDDIGITAL
U1
VDD
R1
470
PIC16C74
GNDDIGITAL
D1
1N4148
10K
R2
3.3uF
C7
S1
GNDDIGITAL
R5
330
R4
330
R3
330
VERDE
D2
AMARILLO
D3
ROJO
D4
GNDDIGITAL
VDD
P1:1
P1:2
IDC2X13F
IDC2X13F
GNDDIGITAL
VDD
U2
C3
22uF
C4
22uF
MAX232
GNDDIGITAL
22uF
C5
22uF
C6
GNDDIGITAL
P2
DB9M
VDD
0.1uF
C8
0.1uF
C9
CONN10M
P3.1
CONN10M
P3.2
GNDDIGITAL
Title
INTERFAZ CON PIC16C74
Size
B
Number
LAB ELECTRONICA
CENTRO DE INSTRUMENTOS, UNAM
Rev
1
Date
Wed Nov 12 1997 15:18:54
Drawn By
MABS
Filename
PICMAN2.SCH
Sheet
1 of 1

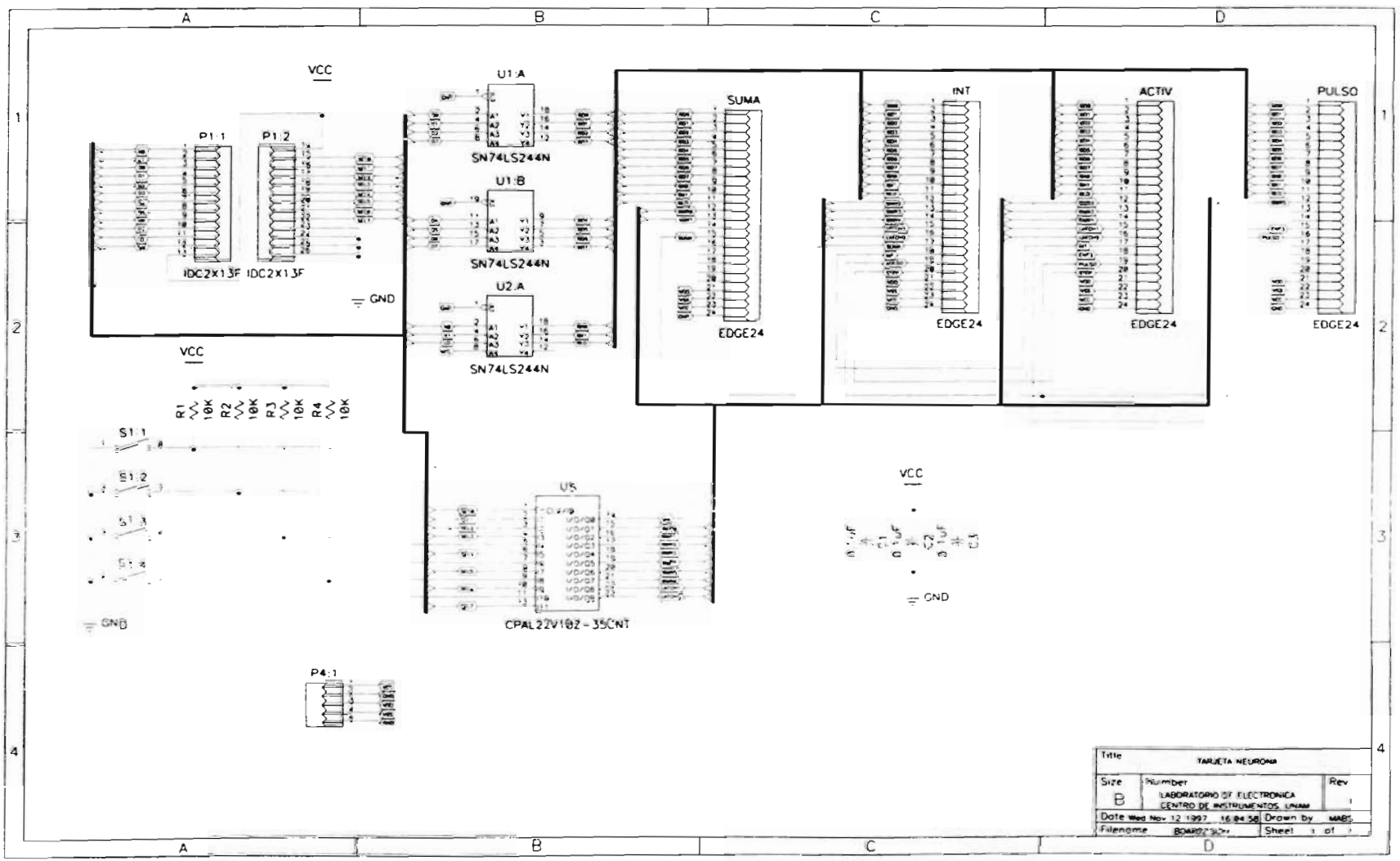




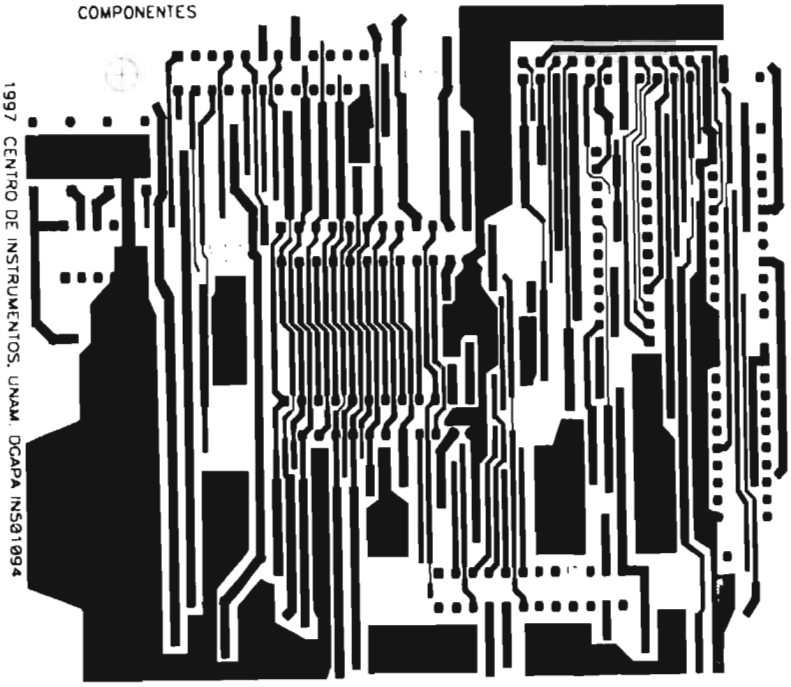

P3
FUENTE
V+V- LEDS
P2 RS232
C2
X1
S1
R2
D1
R1
C7
U1
C9
C8
C5
C4
C6
C3
P1

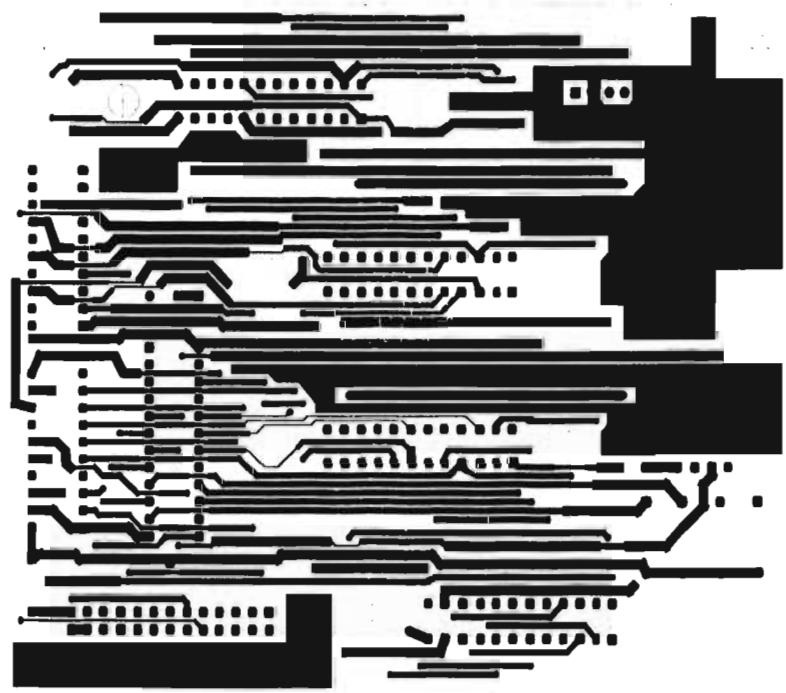


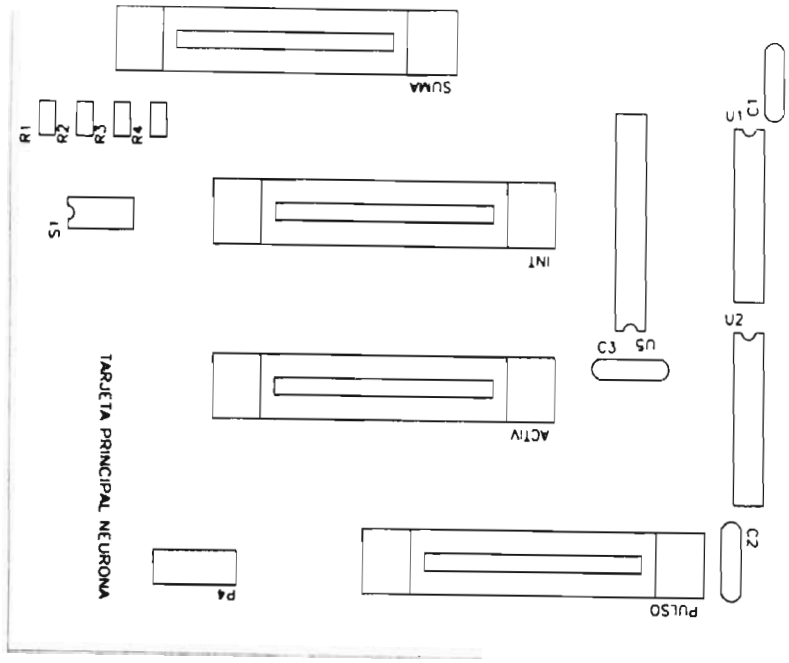
TARJETA BASE

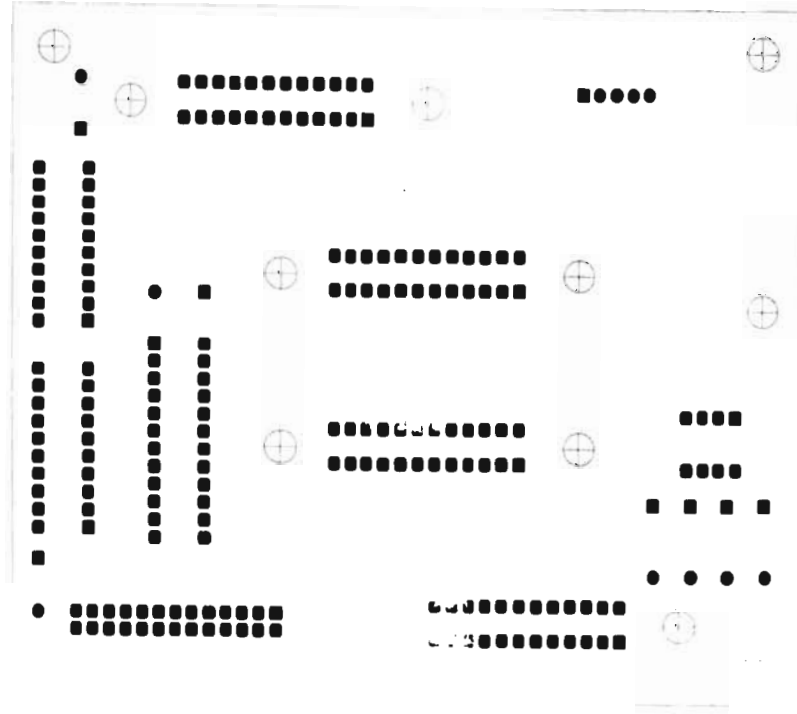

A
B
C
D
1
2
3
4
VCC
P1:1
P1:2
IDC2X13F
IDC2X13F
GND
U1:A
SN74LS244N
U1:B
SN74LS244N
U2:A
SN74LS244N
SUMA
EDGE24
INT
EDGE24
ACTIV
EDGE24
PULSO
EDGE24
R1 10K
R2 10K
R3 10K
R4 10K
S1:1
S1:2
S1:3
S1:4
U5
CPAL22V10Z-35CNT
C1 0.1uF
C2 0.1uF
C3
P4:1
Title TARJETA NEURONA
Size B
Number LABORATORIO DE ELECTRONICA CENTRO DE INSTRUMENTOS UNAM
Rev 1
Date Wed Nov 12 1997 16:04:58
Drawn by MABS
Filename BOARD.SCH
Sheet 1 of 1

COMPONENTES

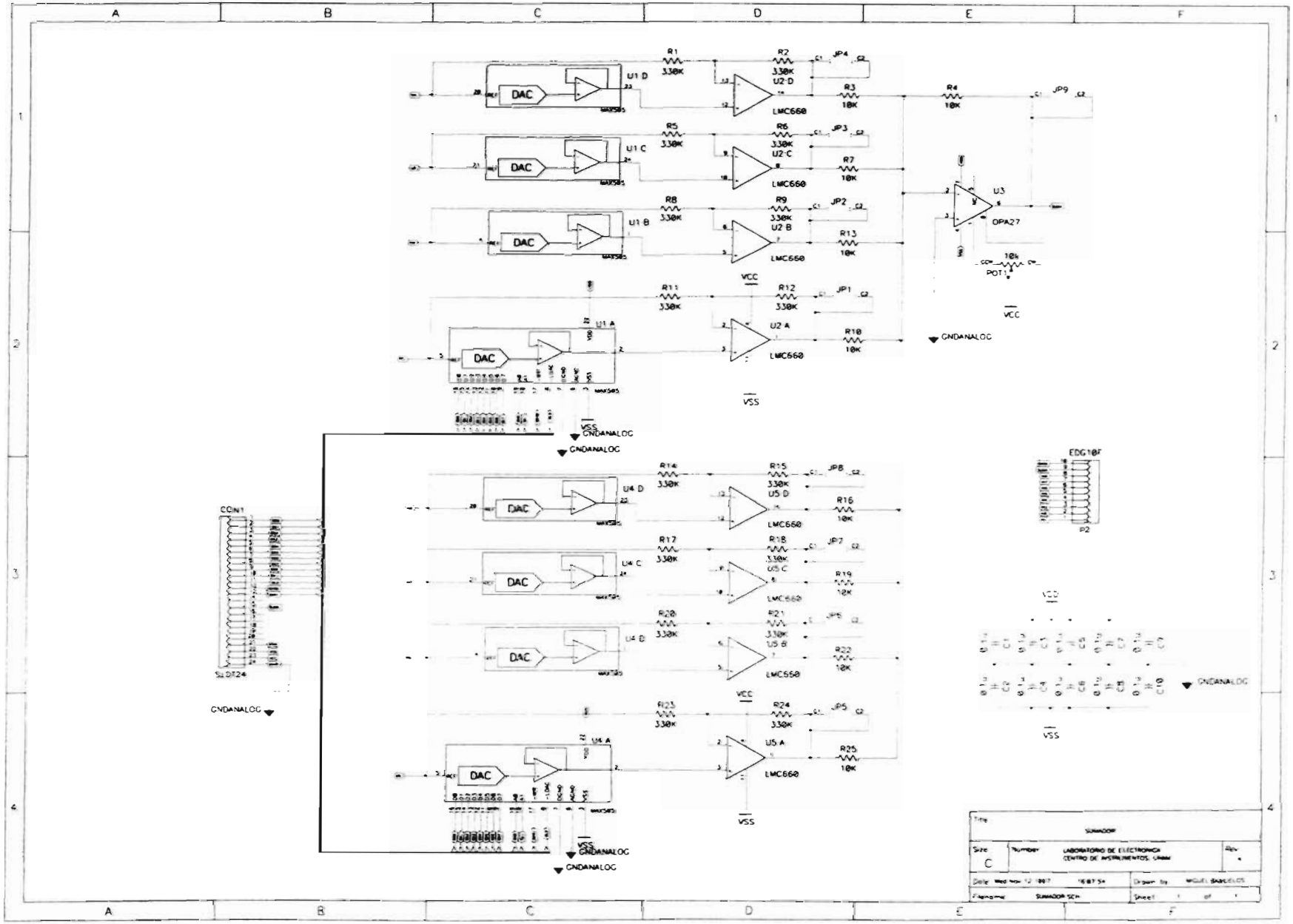


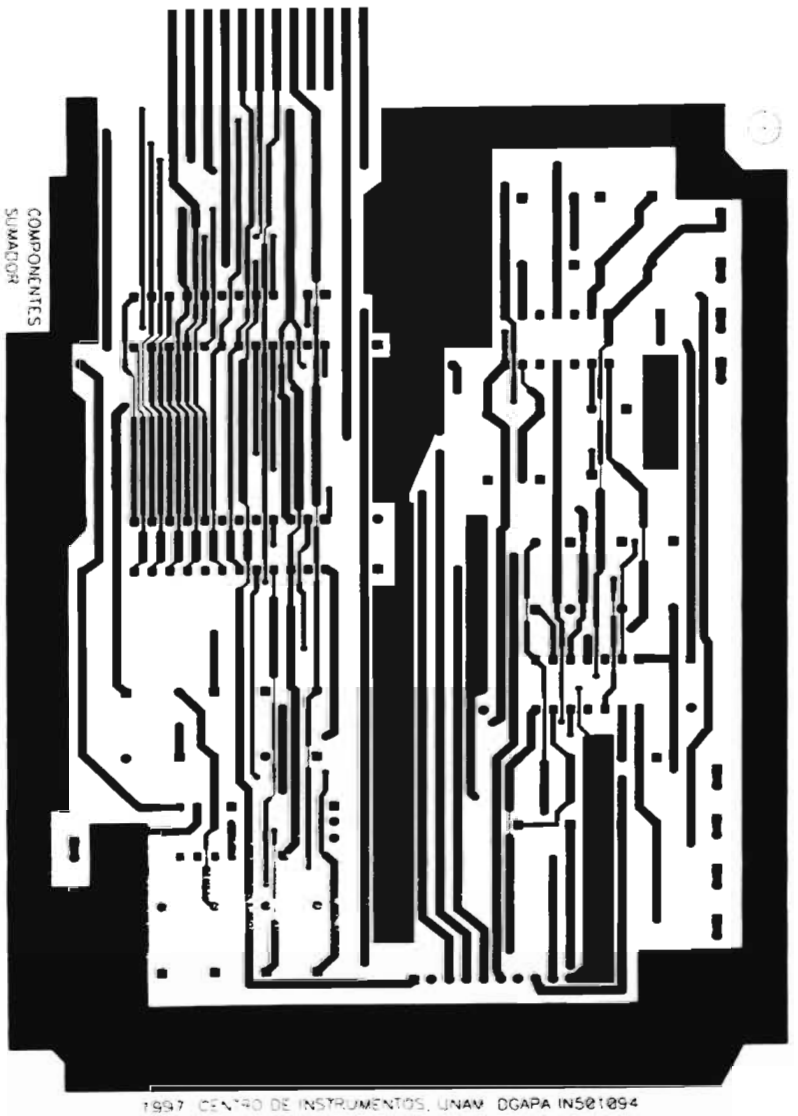


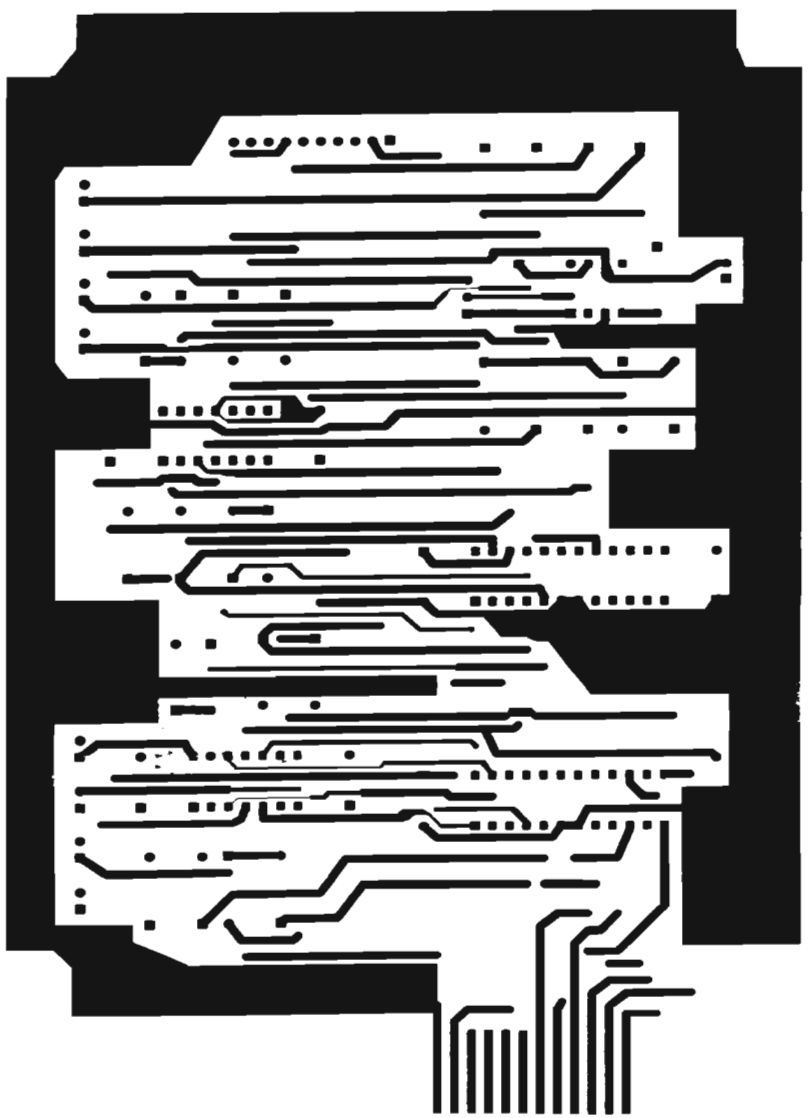

SUMA
R1
R2
R3
R4
S1
INT
ACTIV
TARJETA PRINCIPAL NEURONA
P4
PULSO
C3
U5
U1
C1
U2
C2

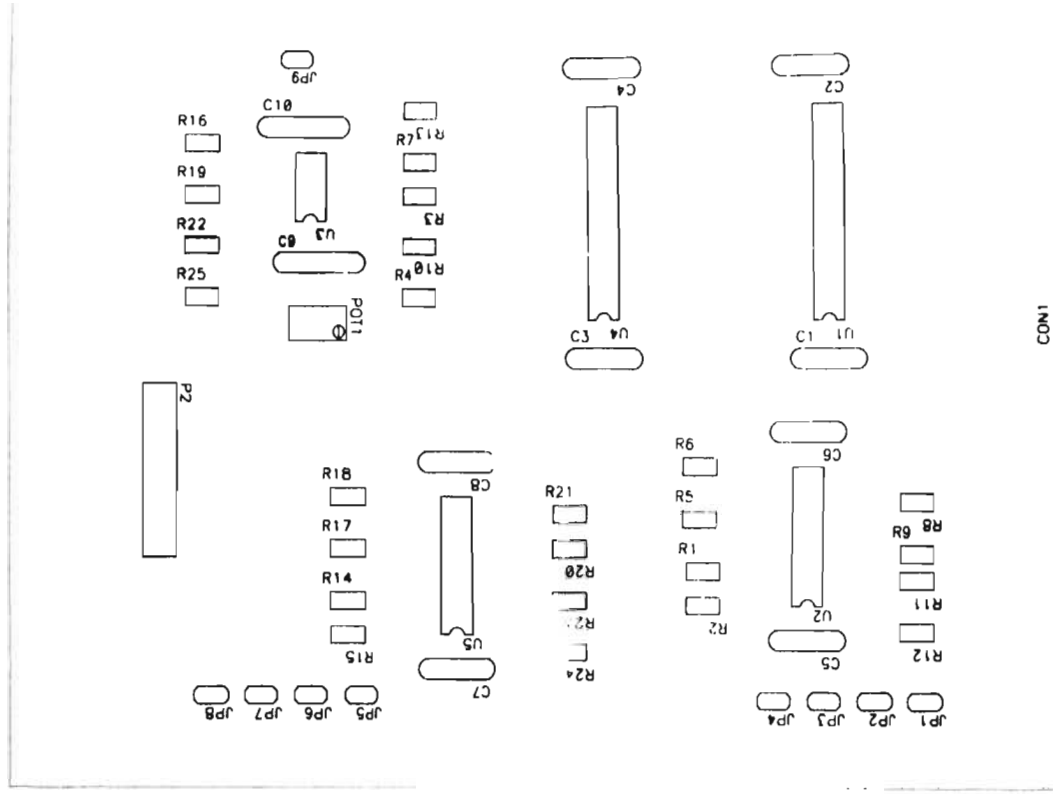


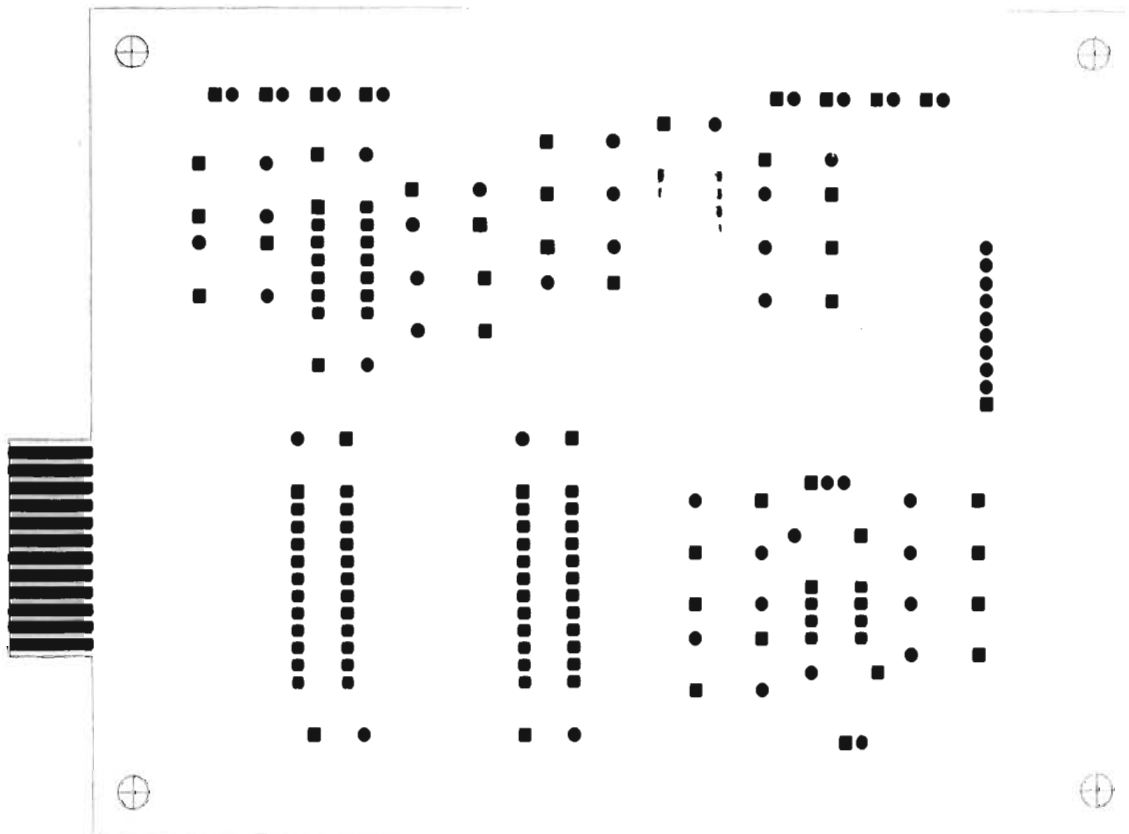
SUMADOR


A
B
C
D
E
F
1
2
3
4
DAC
U1 D
U1 C
U1 B
U1 A
U4 D
U4 C
U4 B
U4 A
LMC660
U2 D
U2 C
U2 B
U2 A
U5 D
U5 C
U5 B
U5 A
LMC550
OPA27
U3
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
R16
R17
R18
R19
R20
R21
R22
R23
R24
R25
330K
10K
JP1
JP2
JP3
JP4
JP5
JP6
JP7
JP8
JP9
POT1
10k
VCC
VSS
GNDANALOG
CON1
SUBT24
EDG10F
P2
Title
SUMADOR
Size
C
Number
LABORATORIO DE ELECTRONICA
CENTRO DE INSTRUMENTOS, UNAM
Rev
Date: Wed Nov 12 1997
16:07:54
Drawn by: MIGUEL BARCELOS
Filename: SUMADOR.SCH
Sheet 1 of 1

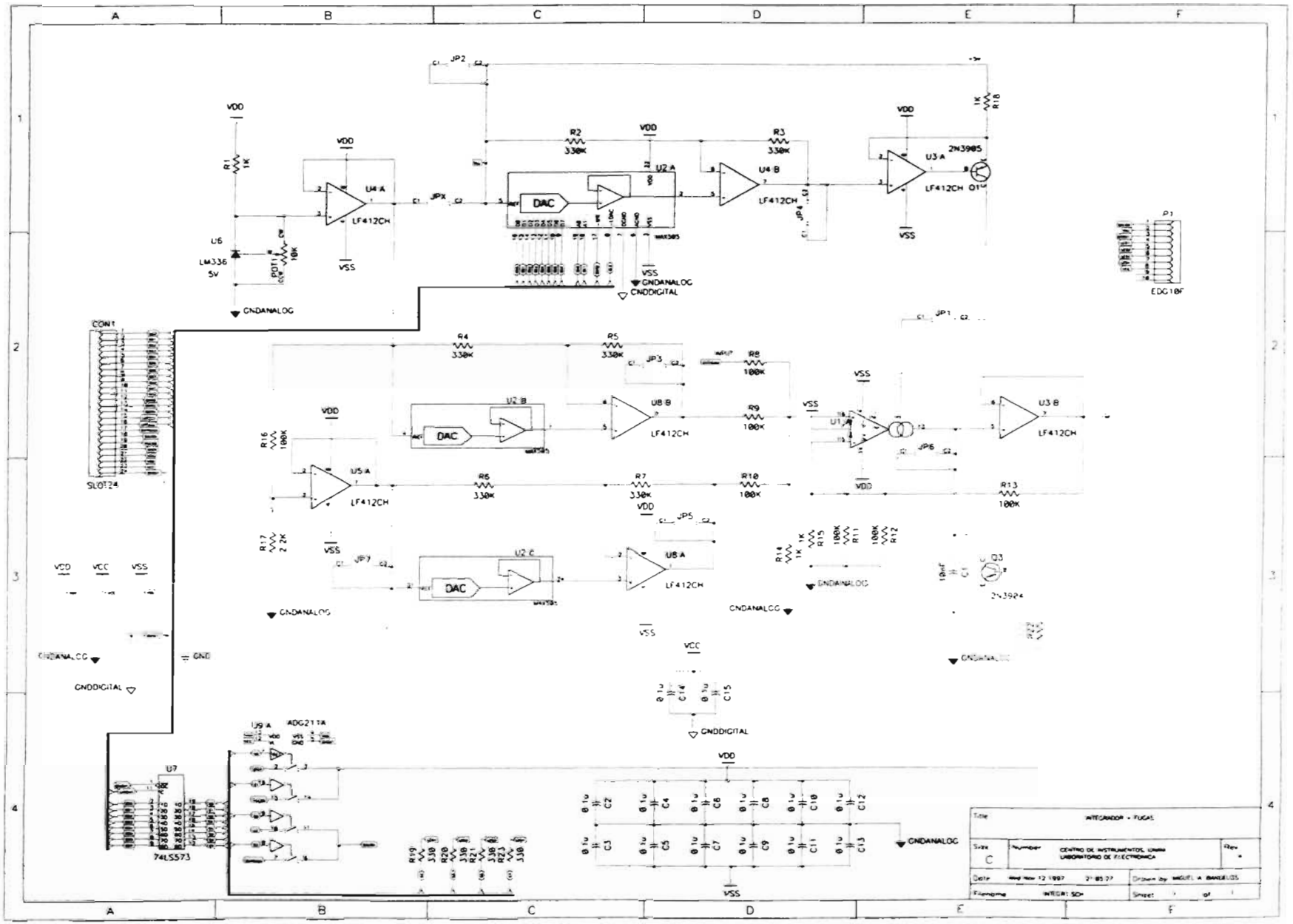

COMPONENTES
SUMADOR
1997 CENTRO DE INSTRUMENTOS, UNAM DGAPA IN501094

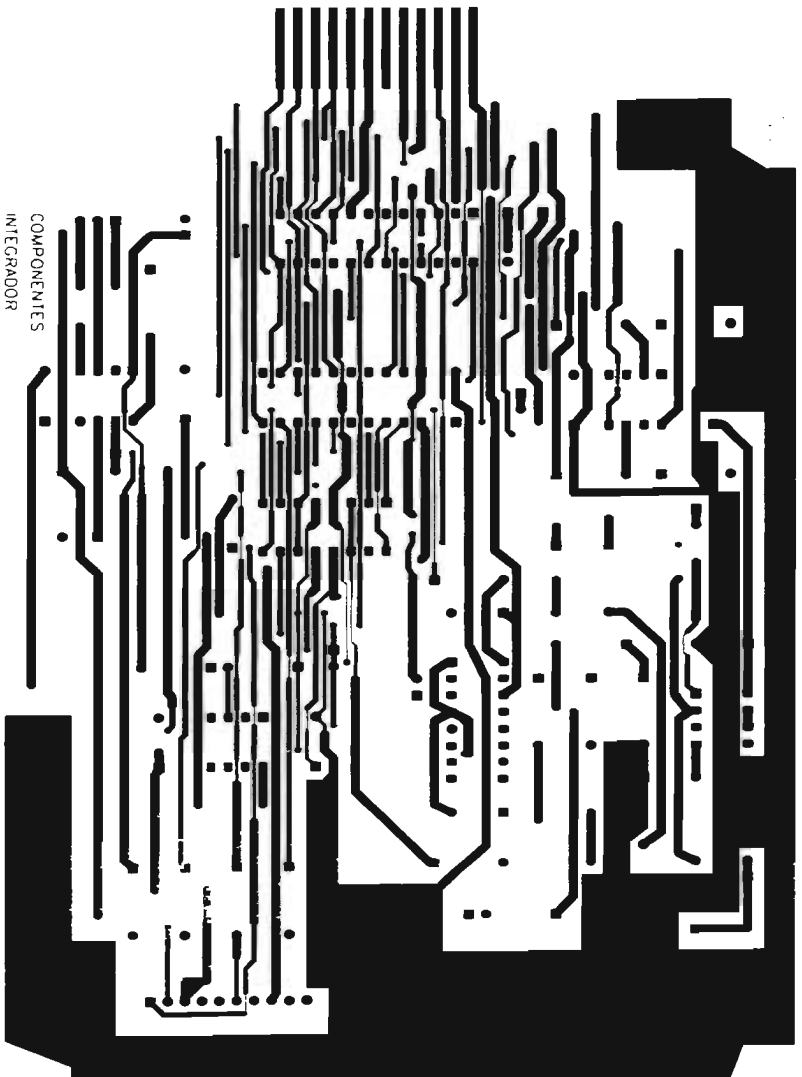


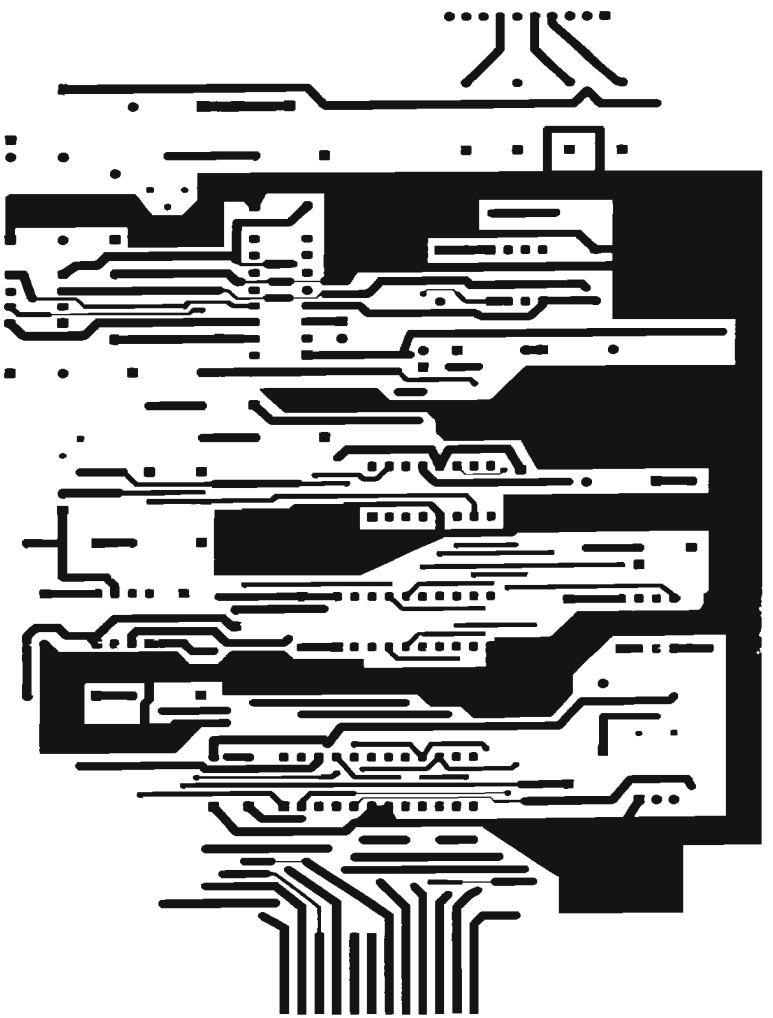

JP9
C10
R16
R19
R22
R25
C9
U3
POT1
R7
R13
R3
R10
R4
C4
C2
U4
C3
U1
C1
CON1
P2
R18
R17
R14
R15
C8
U5
C7
R21
R20
R24
R6
R5
R1
R2
C6
U2
C5
R9
R8
R11
R12
JP8
JP7
JP6
JP5
JP4
JP3
JP2
JP1

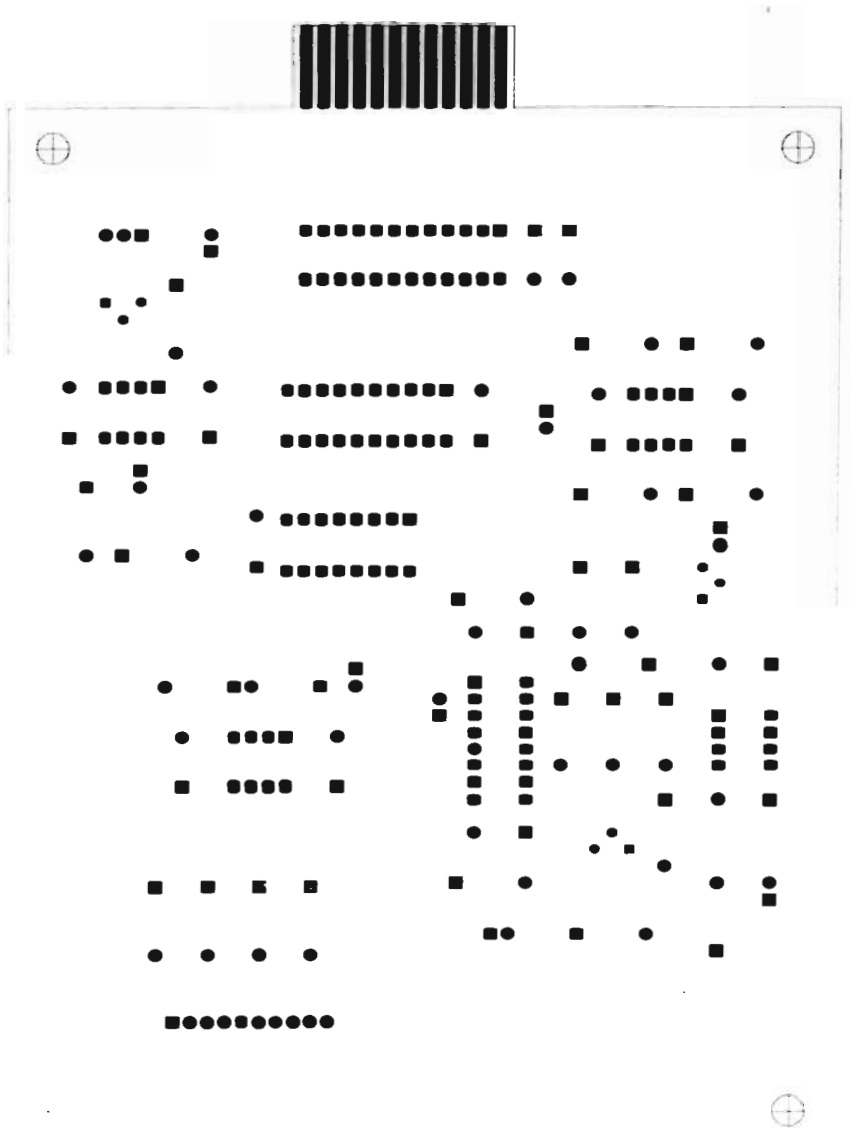


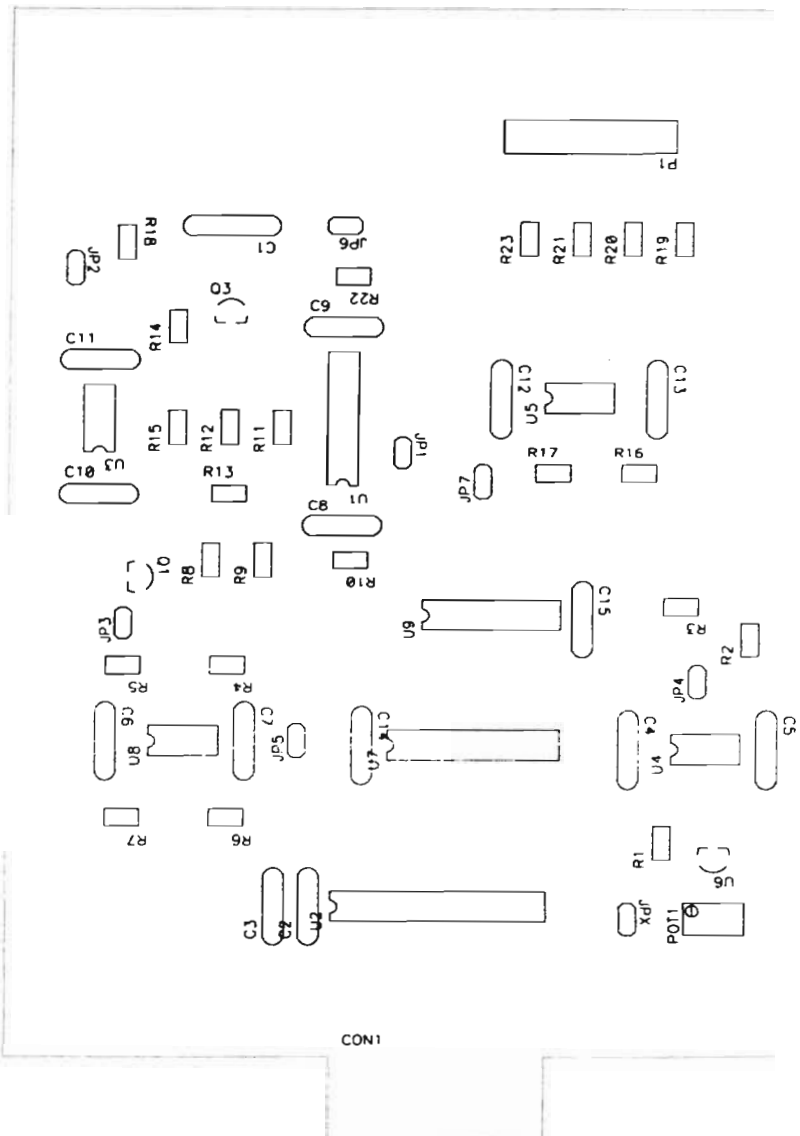
INTEGRADOR


VDD
R1 1K
U6
LM336 5V
POT1 10K
U4 A
LF412CH
VSS
GNDANALOG
JP2
R2 330K
R3 330K
U2 A
DAC
U4 B
LF412CH
U3 A
2N3905
Q1
R18 1K
P1
EDG10F
GNDDIGITAL
CON1
SLOT24
R4 330K
R5 330K
JP3
R8 100K
U2 B
U8 B
R9 100K
U1
JP1
JP6
U3 B
R16 100K
U5 A
R6 330K
R7 330K
R10 100K
R13 100K
JP5
R17 2.2K
JP7
U2 C
U8 A
R14 1K
R15 100K
R11 100K
R12 100K
C1 10nF
Q3
2N3904
VCC
VSS
GND
C14 0.1u
C15 0.1u
U9 A
ADG211A
U7
74LS573
R19 330
R20 330
R21 330
R22 330
R23 330
C2 0.1u
C4 0.1u
C6 0.1u
C8 0.1u
C10 0.1u
C12 0.1u
C3 0.1u
C5 0.1u
C7 0.1u
C9 0.1u
C11 0.1u
C13 0.1u
Title
INTEGRADOR + FUGAS
Size
C
Number
CENTRO DE INSTRUMENTOS UNAM
LABORATORIO DE ELECTRONICA
Rev
Date
Wed Nov 12 1997
21:05:27
Drawn by MIGUEL A RAMIREZ
Filename
INTEGR1.SCH
Sheet
of

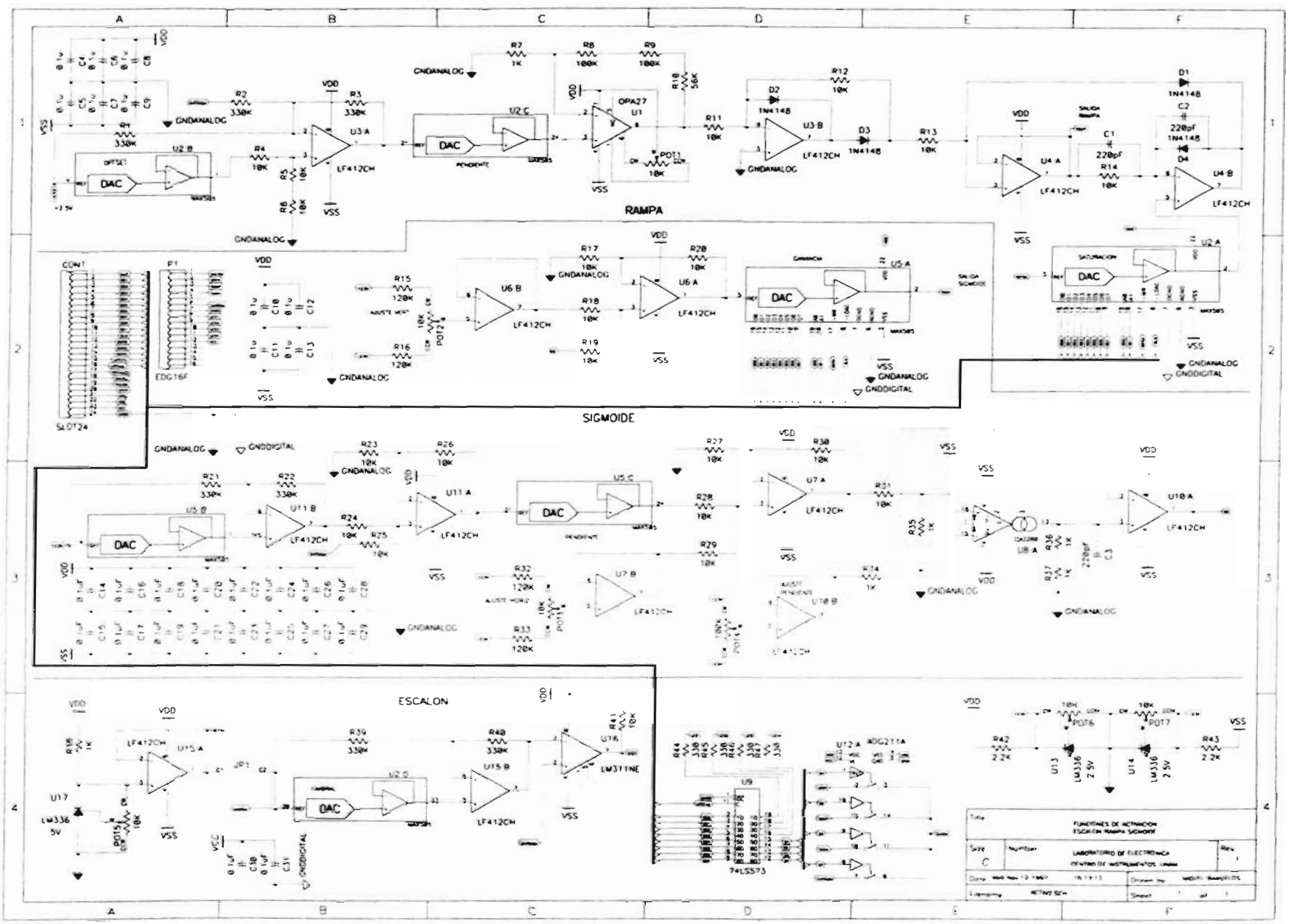


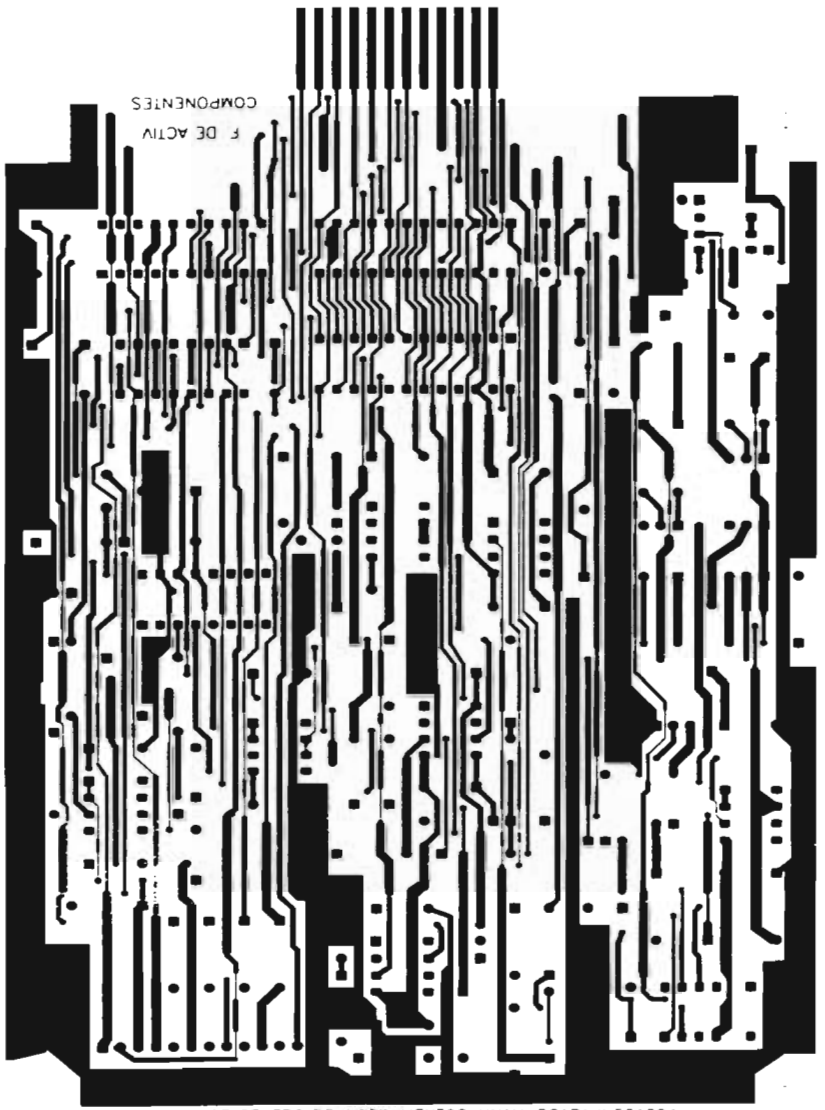


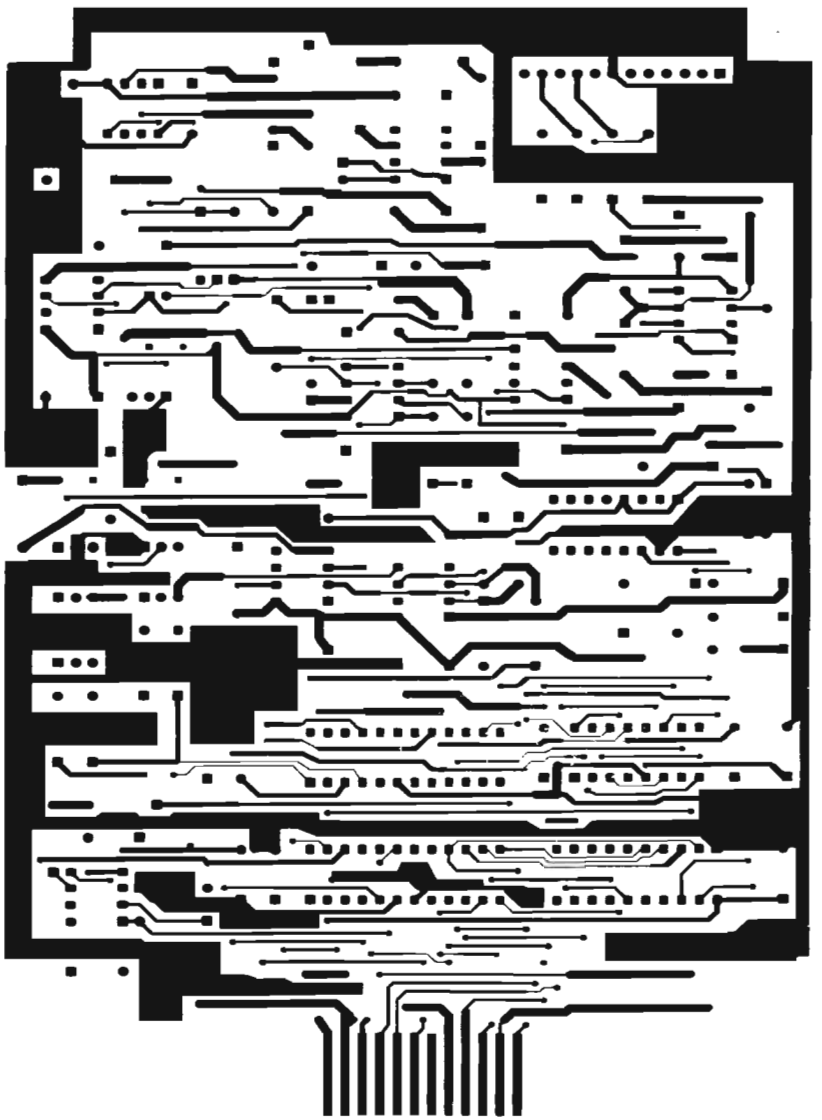


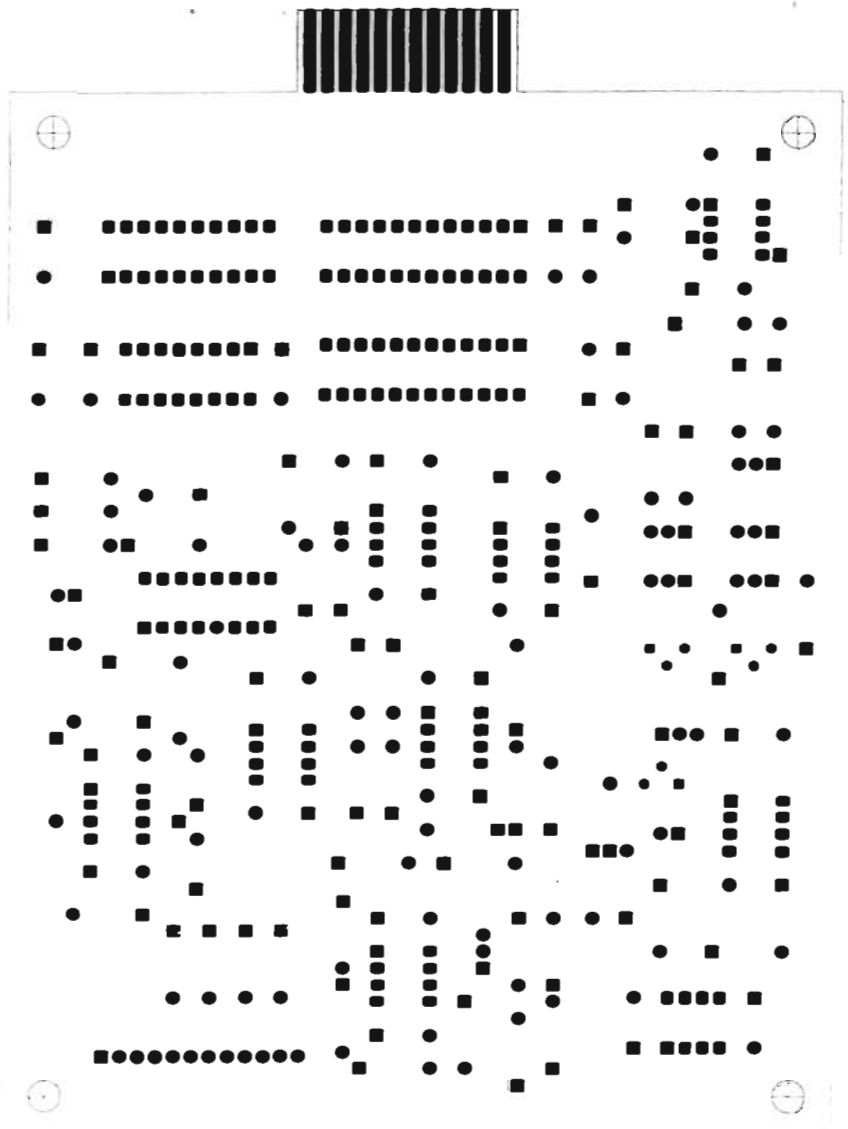

P1
R18
C1
JP6
JP2
R22
Q3
C9
R14
C11
R23
R21
R20
R19
C12
U5
C13
R15
R12
R11
JP1
JP7
R17
R16
C10
U3
R13
U1
C8
R10
Q1
R8
R9
C15
U9
R3
JP3
R5
R4
R2
JP4
C6
C7
JP5
U8
U7
C14
U4
C5
R7
R6
R1
U6
C3
C2
U2
JP8
POT1
CON1

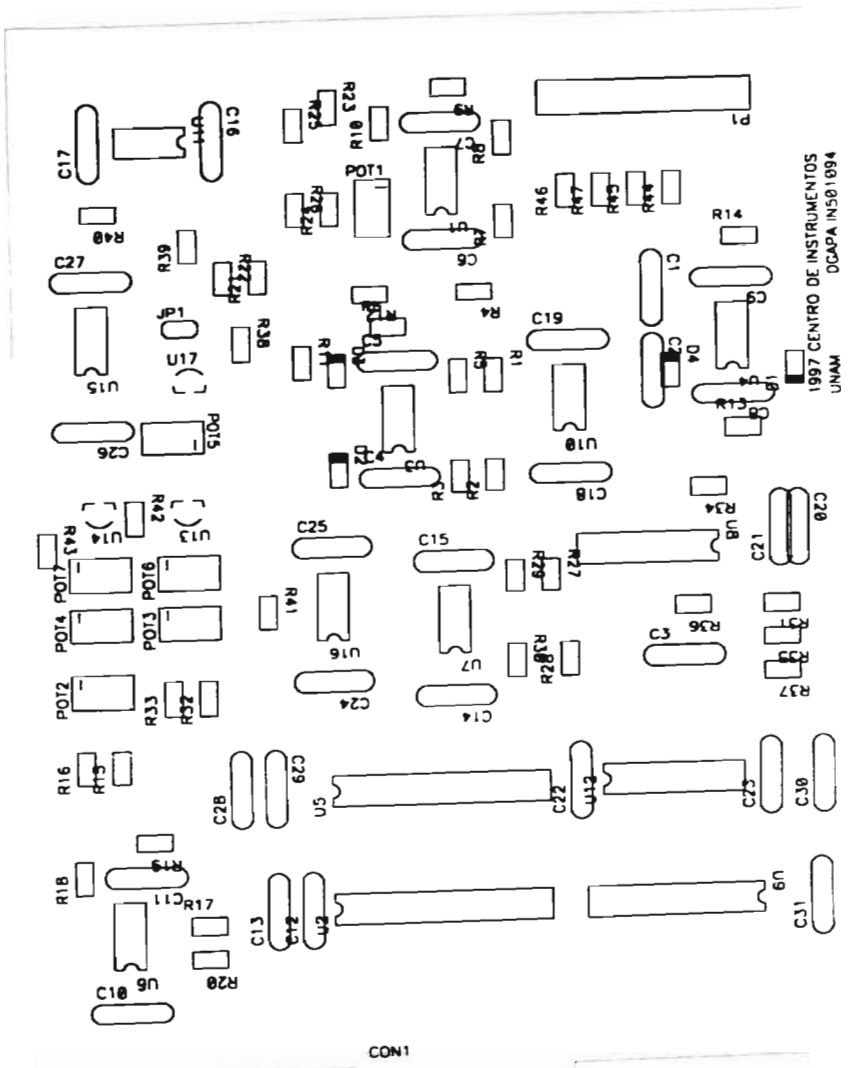
FUNCIONES DE ACTIVACION


RAMPA
SIGMOIDE
ESCALON
DAC
LF412CH
OPA27
74LS573
LM336
2.5V
5V
EDG16F
SLOT24
FUNCIONES DE ACTIVACION
ESCALON RAMPA SIGMOIDE
LABORATORIO DE ELECTRONICA
CENTRO DE INSTRUMENTOS UNAM

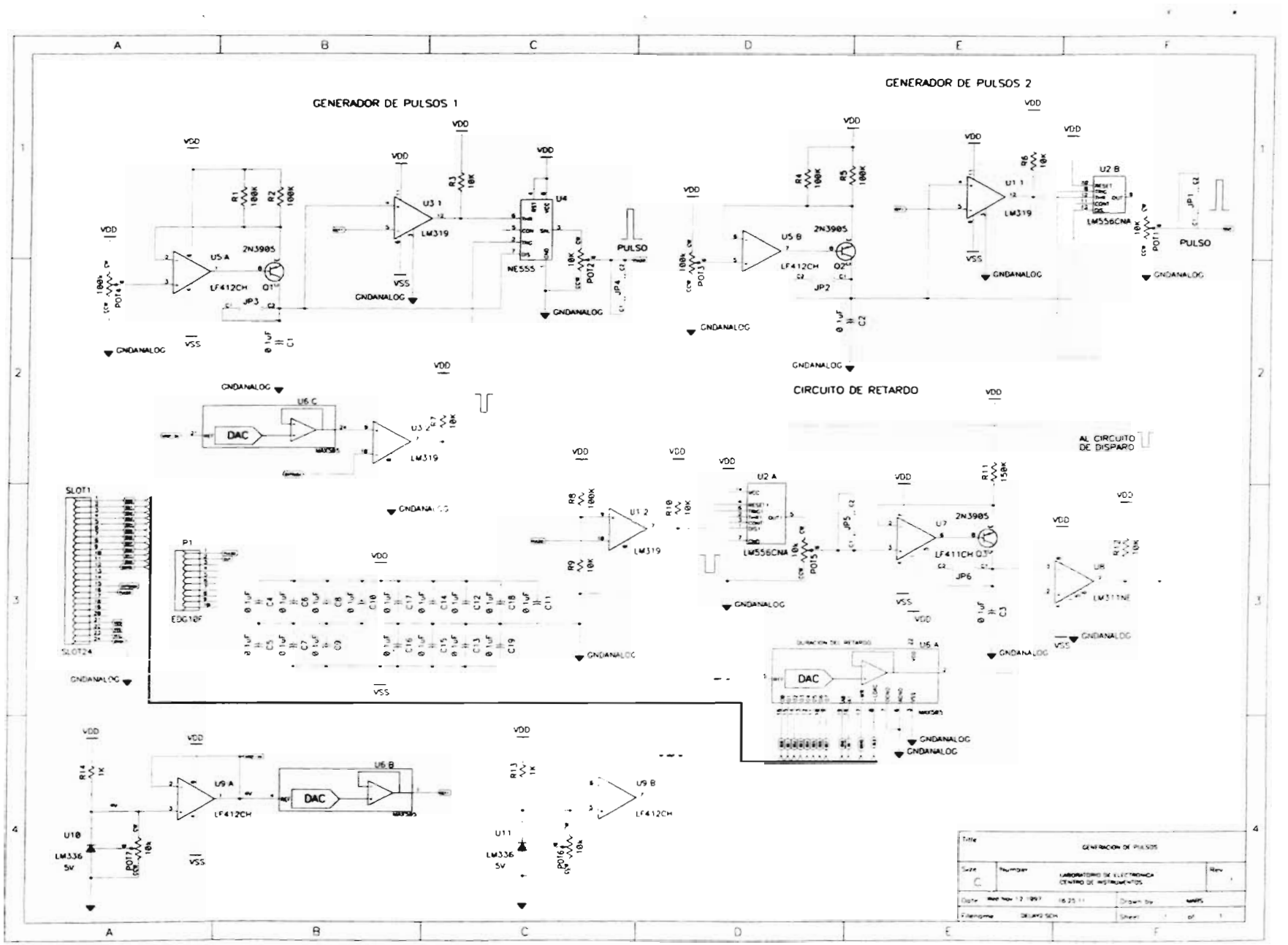




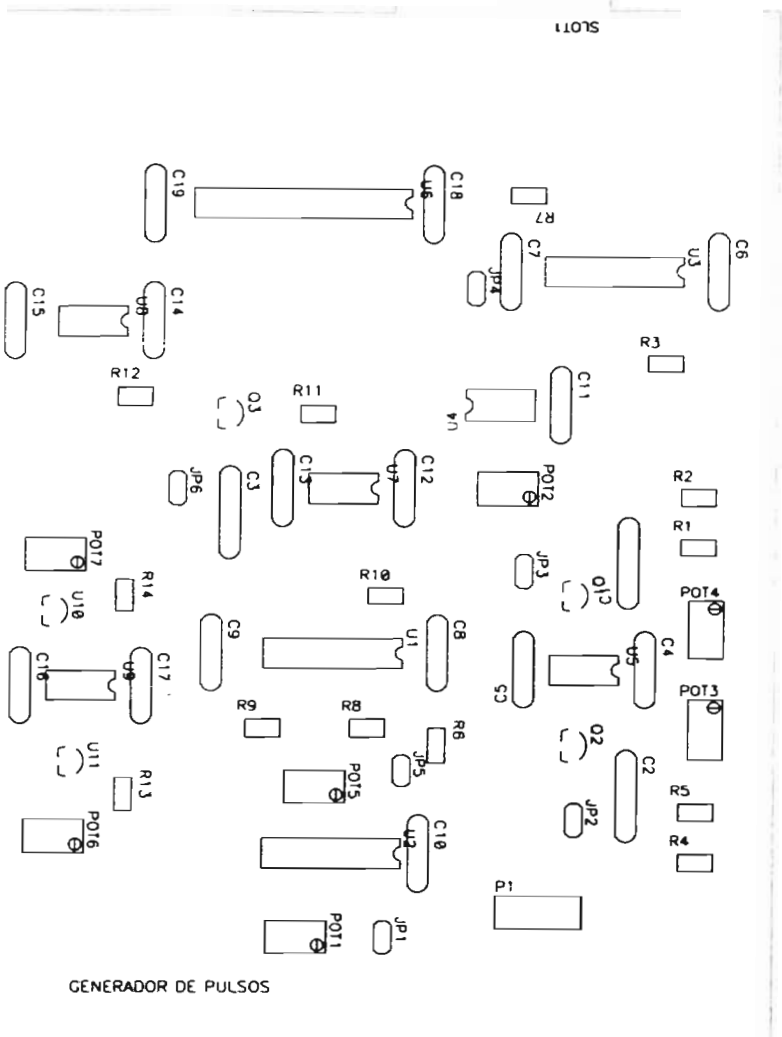



1997 CENTRO DE INSTRUMENTOS
UNAM
DGAPA IN501094
CON1

GENERADOR DE PULSOS


GENERADOR DE PULSOS 1
GENERADOR DE PULSOS 2
CIRCUITO DE RETARDO
AL CIRCUITO DE DISPARO
PULSO
PULSO
DAC
DAC
DAC
LM319
LM319
LM319
LF412CH
LF412CH
LF412CH
LF411CH
2N3905
2N3905
2N3905
NE555
LM556CNA
LM556CNA
LM311NE
LM336 5V
LM336 5V
EDG10F
SLOT1
SLOT24
VDD
VSS
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GNDANALOG
GENERACION DE PULSOS
LABORATORIO DE ELECTRONICA
CENTRO DE INSTRUMENTOS

COMPONENTES
GEN. DE PULSOS


SLOT1
C19
C18
U6
R7
C7
JP4
U3
C6
C15
C14
U8
R3
R12
Q3
R11
C11
U4
JP6
C3
C13
U7
C12
POT2
R2
R1
POT7
JP3
Q1
R14
U10
R10
POT4
C9
U1
C8
C4
U5
C10
C16
U9
C17
C5
POT3
R9
R8
R6
Q2
U11
JP5
C2
R13
POT5
JP2
R5
POT6
U2
C10
R4
P1
POT1
JP1
GENERADOR DE PULSOS

APENDICE II

LISTA DE PARTES

LISTA DE PARTES

INTERFAZ PIC16C74

CANT.	PARTE	DESCRIPCION
2	C1 C2	Cap. Cer. 15pF @ 50V
4	C3 C4 C5 C6	Cap. Electrolítico radial 22μF @ 16V
1	C7	Cap. Electrolítico axial 3.3μF @ 16V
2	C8 C9	Cap. Cer. 0.1μF @ 50V
1	R1	Res. 470Ω @ ½ W 5%
1	R2	Res. 10kΩ @ ½ W 5%
3	R3 R4 R5	Res. 330Ω @ ½ W 5%
1	P1	Conector alpha 26pos.
1	P2	Conector EIS 5 pos. vertical
1	P3	Conector EIS 5 pos. vertical
1	U1	Microcontrolador PIC16C74
1	U2	C.I. MAX232 manejador de interfaz serial
1	D1	Diodo 1N4148
1	D2	LED Verde
1	D3	LED Amarillo
1	D4	LED Rojo
1	X1	Cristal 20.000 Mhz
1	S1	Interruptor de presión
1	U1	Base CI 40 pines
1	U2	Base CI 16 pines

TARJETA BASE

CANT.	PARTE	DESCRIPCION
3	C1 C2 C3	Cap. Cer. 0.1µF @ 50V
1	P1	Conector alpha 26pos.
1	P4	Conector EIS 5 pos. horizontal
2	U1 U2	C.I. 74LS244 Buffer octal
1	U5	GAL22V10
2	S1	Dipswitch 4 pos.
1	SUMA INT ACTIV PULSO	Conector tipo peine 24 pos.
4	R1 R2 R3 R4	Res. 10kΩ @ ½ W 5%
2	U1 U2	Bases C.I. 20 pines
1	U5	Base (8pines + 16pines)

SUMADOR

CANT.	PARTE	DESCRIPCION
10	C1-C10	Cap. Cer. 0.1 µF @ 50V
16	R1 R2 R5 R6 R8 R9 R11 R12 R14 R15 R17 R18 R20 R21 R23 R24	Res. 330kΩ @ ½ W 1%
9	R3 R4 R7 R10 R13 R16 R19 R22 R25	Res. 10kΩ @ ½ W 1%
1	POT1	Trimpot 10k
1	P2	Conector EIS 10 pos. horizontal
2	U1 U4	C.I. MAX505
2	U2 U5	C.I. LMC660
1	U3	C.I. OPA27
9	JP1-JP9	Jumper
3		Base C.I. 8 pines
2		Base C.I. 14 pines
2		Base C.I. 16 pines

INTEGRADOR

CANT.	PARTE	DESCRIPCION
1	C1	Cap. Cer. 10nF @ 50V
14	C2-C15	Cap. Cer. 0.1µF @ 50V
1	R1	Res. 1kΩ @ ½ W 5%
6	R2-R7	Res. 330kΩ @ ½ W 1%
7	R8-R13 R16	Res. 100kΩ @ ½ W 1%
3	R14 R15 R18	Res. 1kΩ @ ½ W 1%
1	R17	Res. 2.2kΩ @ ½ W 1%
4	R19 R20 R21 R23	Res. 330Ω @ ½ W 5%
1	R22	Res. 330Ω @ ½ W 5%
1	U1	C.I. CA3280
1	U2	C.I. MAX505
4	U3 U4 U5 U8	C.I. LF412
1	U6	C.I. LM336-5V
1	U7	C.I. 74LS573
1	U9	C.I. ADG211A
1	P1	Conector EIS 10 pos. horizontal
1	POT1	Trimpot 10k
1	Q1	2N3905
1	Q3	2N3904
7	JP1-JP7	Jumper
1	JPX	Conector EIS 2 pos. horizontal
1		Mini switch 2P2T
5		Base C.I. 8 pines
3		Base C.I. 16 pines
1		Base C.I. 20 pines

FUNCIONES DE ACTIVACION

CANT.	PARTE	DESCRIPCION
3	C1 C2 C3	Cap. Cer. 220pF @ 50V
28	C4-C31	Cap. Cer. 0.1 µF @ 50V
1	P1	Conector EIS 12 pines horizontal
1	U1	C.I. OPA27
1	U2 U5	C.I. MAX505
7	U3 U4 U6 U7 U10 U11 U15	C.I. LF412
1	U8	C.I. CA3280
1	U9	C.I. 74LS573
1	U12	C.I. ADG211A
2	U13 U14	C.I. LM336-2.5V
1	U16	C.I. LM311
1	U17	C.I. LM336-5V
4	D1 D2 D3 D4	Diodo 1N4148
6	POT1 POT2 POT3 POT5 POT6 POT7	Trimpot 10kΩ
1	POT4	Trimpot 100kΩ
7	R1 R2 R3 R21 R22 R39 R40	Res. 330kΩ @ ½ W 1%
20	R4 R5 R6 R11 R12 R13 R14 R17 R18 R19 R20 R23 R24 R25 R26 R27 R28 R29 R30 R31	Res. 10kΩ @ ½ W 1%
1	R41	Res. 10kΩ @ ½ W 5%
5	R7 R34 R35 R36 R37	Res. 1kΩ @ ½ W 1%
1	R38	Res. 1kΩ @ ½ W 5%
2	R8 R9	Res. 100kΩ @ ½ W 1%
1	R10	Res. 56kΩ @ ½ W 1%
4	R15 R16 R32 R33	Res. 120kΩ @ ½ W 1%
2	R42 R43	Res. 2.2kΩ @ ½ W 5%
4	R44 R45 R46 R47	Res. 330Ω @ ½ W 5%
1	JP1	Conector EIS 2 pos. horizontal
1		Mini interruptor 2P2T
11		Bases C.I. 8 pines
4		Bases C.I. 16 pines
1		Bases C.I. 20 pines

GENERADOR DE PULSOS

CANT	PARTE	DESCRIPCION
3	C1 C2 C3	Cap. Poliester 0.1 µF
16	C4-C19	Cap. Cer. 0.1 µF @ 50V
4	R1 R2 R4 R5	Res. 100kΩ @ ½ W 1%
6	R3 R6 R7 R9 R10 R12	Res. 10kΩ @ ½ W 5%
1	R8	Res. 100kΩ @ ½ W 5%
1	R11	Res. 150kΩ @ ½ W 1%
2	R13 R14	Res. 1kΩ @ ½ W 5%
5	POT1 POT2 POT5 POT6 POT7	Trimpot 10k
2	POT3 POT4	Trimpot 100k
3	Q1 Q2 Q3	2N3905
1	P1	Conector EIS 5pines horizontal
2	U1 U3	C.I. LM319
1	U2	C.I. LM556
1	U4	C.I. LM555
2	U5 U9	C.I. LF412
1	U6	C.I. MAX505
1	U7	C.I. LF411
1	U8	C.I. LM311
2	U10 U11	C.I. LM336-5V
6	JP1-JP6	Jumper
6		Bases C.I. 8 pines
3		Bases C.I. 14 pines
1		Base C.I. 16 pines

APENDICE III

LISTADO DE PROGRAMAS

PROGRAMA EN ENSAMBLADOR PIC16C74

```
;*********************************************************
*************
;*** PROGRAMA DE INTERFAZ SERIAL PARA LA
NEUROCOMPUTADORA ***
;*********************************************************
*************

        LIST    p=16C74A

        #DEFINE RCIF 0x0C,5
        #DEFINE TXIF 0x0C,4

INDF    EQU     00
STATUS  EQU     03
FSR     EQU     04
PTOA    EQU     05
PTOB    EQU     06
PTOC    EQU     07
PTOD    EQU     08
PTOE    EQU     09
TRISA   EQU     85
TRISB   EQU     86
TRISC   EQU     87
TRISD   EQU     88
TRISE   EQU     89
ADCON1  EQU     9F
TXSTA   EQU     98
RCSTA   EQU     18
SPBRG   EQU     99
TXREG   EQU     19
RCREG   EQU     1A
PCL     EQU     02
PESO1   EQU     20
PESO2   EQU     21
PESO3   EQU     22
PESO4   EQU     23
PESO5   EQU     24
PESO6   EQU     25
PESO7   EQU     26
PESO8   EQU     27
INTTIP  EQU     28
DISPAR  EQU     29
TAU     EQU     2A
POT     EQU     2B
OFFSET  EQU     2C
ACTTIP  EQU     2D
UMBRAL  EQU     2E
PEND    EQU     2F
SAT     EQU     30
CORR    EQU     31
PENDS   EQU     32
SATS    EQU     33
CORRS   EQU     34
PULSO   EQU     35
RETARD  EQU     36
IX      EQU     50
IY      EQU     51
ALFA    EQU     52

        ORG 0000

        GOTO 0005

        ORG 0005

        ;************** INICIALIZACION
**********************

INI:    MOVLW 0x00
        BSF STATUS,5    ;SELECCIONA BANCO 1
        MOVWF TRISA     ;PTO A=SALIDA
        MOVWF TRISB     ;PTO B=SALIDA
        MOVWF TRISC     ;PTO C=SALIDA
        MOVWF TRISD     ;PTO D=SALIDA

        BSF PTOC,0      ;LED VERDE=ON
        BCF PTOC,1      ;LED AMARILLO=OFF
        BCF PTOC,2      ;LED ROJO=OFF

        BSF ADCON1,0    ;PTO E.0=DIGITAL
        BSF ADCON1,1    ;PTO E.1=DIGITAL
        BCF TRISE,0     ;PTO E.0=SALIDA
        BCF TRISE,1     ;PTO E.1=SALIDA
        BCF STATUS,5    ;SELECCIONA BANCO 0
        MOVLW 0x00
        MOVWF PTOD      ;DATOS IGUAL A CERO
        BSF PTOE,0      ;LD=1
        BSF PTOE,1      ;WR=1
        MOVWF PTOA      ;A0=0  A1=0
        MOVWF PTOB      ;DECOD=0

        MOVLW 0x81
        BSF STATUS,5    ;SELECCIONA BANCO 1
        MOVWF SPBRG     ;SELECCIONA 2400
BAUDS
        BCF TXSTA,4     ;SYNC=0 MODO
ASINCRONO
        BSF TXSTA,5     ;TXEN=1 HABILITA
TRANSMISION
        BCF STATUS,5    ;SELECCIONA BANCO 0
        BSF RCSTA,7     ;SPEN=1 HABILITA EL
PUERTO SERIAL
        BSF RCSTA,4     ;CREN=1 HABILITA
RECEPCION

        ;********** INICIO DE COMUNICACION
**************

INICIO: BTFSS RCIF
        GOTO INICIO     ;ESPERA A RECIBIR UN
DATO
        MOVF RCREG,0    ;CARGA EL DATO EN W
        SUBLW 0x49
        BTFSS STATUS,2
        GOTO INICIO     ;ESPERA A RECIBIR UNA
I
```

```
        MOVLW 0x4B
        MOVWF TXREG     ;MANDA UNA K
ETIQ1:  BTFSS TXIF
        GOTO ETIQ1      ;ESPERA EL FIN DE
TRANSMISION

        ;********** RECEPCION DEL COMANDO
**************

CMND:   BTFSS RCIF
        GOTO CMND       ;ESPERA A RECIBIR UN
DATO
        MOVF RCREG,0    ;CARGA EL COMANDO
        ADDLW 0x80      ;SUMA OFFSET DE LA
TABLA
        MOVWF PCL       ;SALTA A LA TABLA

        GOTO INICIO


        ;********** TABLA DE COMANDOS
******************

        ORG 0080

        GOTO DATOS

        GOTO INICIO

        ;************************************
        ;********** RUTINAS **************
        ;************************************

        ORG 0100

        GOTO INICIO

        ;********** DATOS ****************

DATOS:  MOVLW 0x49      ;MANDA UNA I
        MOVWF TXREG
ETIQ2:  BTFSS TXIF
        GOTO ETIQ2      ;ESPERA FIN DE
TRANSMISION

        MOVLW 0x20      ;INICIALIZA REGISTRO
DE DIRECCIONAMIENTO
        MOVWF FSR       ;INDIRECTO CON LA
DIR=20h

        MOVLW 0x0A      ;INICIALIZA EL
REGISTRO 51h
        MOVWF IY        ;COMO CONTADOR DE
NEURONAS

NEUR:   MOVLW 0x23      ;INICIALIZA EL
REGISTRO 50h
        MOVWF IX        ;COMO CONTADOR DE
PARAMETROS

        MOVLW 0x41      ;INICIALIZA EL
REGISTRO 52h
        MOVWF ALFA      ;COMO INICIO DEL
ALFABETO

        ;**** AQUI RECIBE LOS DATOS Y LOS
ALMACENA ***

ETIQ3:  BTFSS RCIF
        GOTO ETIQ3      ;ESPERA A RECIBIR
DATO
        MOVF RCREG,0
        MOVWF INDF      ;GUARDA EL DATO
        INCF INDF,1     ;INCREMENTA
APUNTADOR
        MOVF ALFA,0     ;CARGA ALFA EN EL
ACUMULADOR
        MOVWF TXREG     ;MANDA ALFA
        INCF ALFA,1     ;INCREMENTA EL
ALFABETO
ETIQ4:  BTFSS TXIF
        GOTO ETIQ4      ;ESPERA FIN DE
TRANSMISION
        DECFSZ IX,1     ;DECREMENTA INDICE
DE PARAMETROS
        GOTO ETIQ3


        ;**** AQUI MANDA LOS DATOS A LA
NEUROCOMPUTADORA **

        MOVF IY,0       ;CARGA CONTADOR DE
NEURONAS EN ACC
        SUBLW 0x0A      ;OBTIENE EL NUMERO
DE NEURONA
        MOVWF IY
        BCF STATUS,0    ;LIMPIA EL CARRY
        RLF IY,1  ;ROTA A LA IZQ Y GUARDA EN IY
        BCF STATUS,0    ;LIMPIA EL CARRY
        RLF IY,1  ;ROTA A LA IZQ Y GUARDA EN IY
        BCF STATUS,0    ;LIMPIA EL CARRY
        RLF IY,1  ;ROTA A LA IZQ Y GUARDA EN IY
        BSF IY,7  ;SEL7=1
                  ;AHORA IY TIENE LA
DIRECCION BASE DE LA NEURONA


        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=0  SELECCIONA DAC-
A
        MOVF PESO1
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=0  SELECCIONA DAC-
B
        MOVF PESO2
        CALL MANDA
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=1  SELECCIONA DAC-
C
        MOVF PESO3
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=1  SELECCIONA DAC-
D
        MOVF PESO4
        CALL MANDA

        INCF IY,1 ;INCREMENTA DIRECCION BASE
```

```
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
A
        MOVF PESO5
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
B
        MOVF PESO6
        CALL MANDA
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=1 SELECCIONA DAC-
C
        MOVF PESO7
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=1 SELECCIONA DAC-
D
        MOVF PESO8
        CALL MANDA

        INCF IY,1;INCREMENTA DIRECCION BASE
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
A
        MOVF TAU
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
B
        MOVF POT
        CALL MANDA
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=1 SELECCIONA DAC-
C
        MOVF OFFSET
        CALL MANDA

        INCF IY,1;INCREMENTA DIRECCION BASE
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
A
        MOVF SAT
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
B
        MOVF CORR
        CALL MANDA
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=1 SELECCIONA DAC-
C
        MOVF PEND
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=1 SELECCIONA DAC-
D
        MOVF UMBRAL
        CALL MANDA

        INCF IY,1;INCREMENTA DIRECCION BASE
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
A
        MOVF SATS
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
B
        MOVF CORRS
        CALL MANDA
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=1 SELECCIONA DAC-
C
        MOVF PENDS
        CALL MANDA

        INCF IY,1;INCREMENTA DIRECCION BASE
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE
        BCF PTOA,0      ;A0=0
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
A
        MOVF RETARD
        CALL MANDA
        BCF PTOA,0      ;A0=1
        BCF PTOA,0      ;A1=0 SELECCIONA DAC-
B
        MOVF PULSO
        CALL MANDA

        INCF IY,1;INCREMENTA DIRECCION BASE
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE

        MOVF INTTIP
        BTFSC STATUS,2  ;PREGUNTA SI ES CERO

        GOTO NOINT      ;SIN INTEGRADOR
        DECF INTTIP,1
        BTFSC STATUS,2  ;PREGUNTA SI ES CERO
        GOTO DISP
        MOVLW 0x0DB     ;INTEGRADOR CON
FUGAS
        GOTO ETIQ5

NOINT:  MOVLW 0x0F7     ;SIN INTEGRADOR
        GOTO ETIQ5

DISP:   MOVF DISPAR
        BTFSC STATUS,2  ;PREGUNTA SI ES CERO
        GOTO STEP
        MOVLW 0x69      ;ES PULSO
        GOTO ETIQ5

STEP:   MOVLW 0x0EA     ;ES ESCALON

ETIQ5:  CALL MANDA      ;LATCH INTEGRADOR
```

```
        INCF IY,1 ;INCREMENTA DIRECCION BASE
        MOVF IY,0       ;CARGA DIRECCION
BASE EN ACC
        MOVWF PTOB      ;ENVIA DIRECCION BASE

        MOVF ACTTIP
        BTFSC STATUS,2  ;PREGUNTA SI ES CERO
        GOTO NOACT
        DECF ACTTIP,1
        BTFSC STATUS,2  ;PREGUNTA SI ES CERO
        GOTO ESCAL
        DECF ACTTIP,1
        BTFSC STATUS,2
        GOTO RAMP
        MOVLW 0x0B7     ;ES TANH
        GOTO ETIQ6

NOACT:  MOVLW 0x7E
        GOTO ETIQ6

ESCAL:  MOVLW 0x0ED
        GOTO ETIQ6

RAMP:   MOVLW 0x0DB

ETIQ6:  CALL MANDA


        DECFSZ IY,1     ;DECREMENTA
CONTADOR DE NEURONAS
        GOTO NEUR       ;REPITE PARA LA SIG.
NEURONA

        GOTO INICIO


        ;******** SUBRUTINA MANDA DATO AL
DAC *********
        ;** EL DATO SE ENCUENTRA EN EL
ACUMULADOR W ****

MANDA:  MOVWF PTOD      ;ESCRIBE EL DATO
        BCF PTOE,1      ;WR=0
        BSF PTOE,1      ;WR=1
        BCF PTOE,0      ;LD=0
        BSF PTOE,0      ;LD=1
        RETURN


        END
```

PROGRAMA EN VISUAL BASIC

```
Private Sub DBGrid1_Click()

End Sub



Private Sub MDIForm_Load()

'INICIALIZACION DE VARIABLES

neur = 1

For i = 1 To 10
   inttip(neur) = 0 'sin integrador
   tau(i) = 0
   pot(i) = 128
   offset(i) = 128
   acttip(i) = 0 'sin función de activación
   disparo(i) = 0 'disparo por escalón
   umbral(i) = 128
   pend(i) = 1
   sat(i) = 255
   corr(i) = 128
   pends(i) = 255
   sats(i) = 255
   corrs(i) = 128
   pulso(i) = 255
   retardo(i) = 255
Next i

VScroll1.Value = neur
Label1.Caption = Val(VScroll1.Value)

Call llenagrid
  'Cambia la aplicación a sus propio directorio.
  ChDir App.Path
  ChDrive App.Path
End Sub

Public Sub llenagrid()
Grid1.Col = 0
Grid1.Row = 0
Grid1.Text = ""
Grid1.Col = 1
Grid1.Text = "w1"
Grid1.Col = 2
Grid1.Text = "w2"
Grid1.Col = 3
Grid1.Text = "w3"
Grid1.Col = 4
Grid1.Text = "w4"
Grid1.Col = 5
Grid1.Text = "w5"
Grid1.Col = 6
Grid1.Text = "w6"
Grid1.Col = 7
Grid1.Text = "w7"
Grid1.Col = 8
Grid1.Text = "w8"
Grid1.Col = 9
Grid1.Text = "Integ"
Grid1.Col = 10
Grid1.Text = "Trig"
Grid1.Col = 11
Grid1.Text = "Tau"
Grid1.Col = 12
Grid1.Text = "R.Pot"
Grid1.Col = 13
Grid1.Text = "Offset"
Grid1.Col = 14
Grid1.Text = "Activ"
Grid1.Col = 15
Grid1.Text = "Thres"
Grid1.Col = 16
Grid1.Text = "Slope"
Grid1.Col = 17
Grid1.Text = "Sat"
Grid1.Col = 18
Grid1.Text = "Shift"
Grid1.Col = 19
Grid1.Text = "Pulse"
Grid1.Col = 20
Grid1.Text = "Delay"
Grid1.Col = 0
Grid1.Row = 1
Grid1.Text = "Neur1"
Grid1.Row = 2
Grid1.Text = "Neur2"
Grid1.Row = 3
Grid1.Text = "Neur3"
Grid1.Row = 4
Grid1.Text = "Neur4"
Grid1.Row = 5
Grid1.Text = "Neur5"
Grid1.Row = 6
Grid1.Text = "Neur6"
Grid1.Row = 7
Grid1.Text = "Neur7"
Grid1.Row = 8
Grid1.Text = "Neur8"
Grid1.Row = 9
Grid1.Text = "Neur9"
Grid1.Row = 10
Grid1.Text = "Neur10"

For i = 1 To 10
   For j = 1 To 8
      datos(i, j) = Format(-((peso(i, j)) / 128 - 1), "#0.0##")
   Next j

   If inttip(i) = 0 Then
      datos(i, 9) = "no"
      datos(i, 11) = ""
      datos(i, 12) = ""
      datos(i, 13) = ""
      datos(i, 10) = ""
   End If
   If inttip(i) = 1 Then
      datos(i, 9) = "trigger"
      datos(i, 11) = Format((tau(i) / 255), "#0.0##")
      datos(i, 12) = Format((pot(i) / 128 - 1) * 5, "#0.0##")
      datos(i, 13) = Format(-(offset(i) / 128 - 1), "#0.0##")
      If disparo(i) = 0 Then datos(i, 10) = "step"
      If disparo(i) = 1 Then datos(i, 10) = "pulse"
   End If
   If inttip(i) = 2 Then
```

```
    datos(i, 9) = "leaky"
    datos(i, 11) = Format((tau(i) / 255), "#0.0##")
    datos(i, 12) = Format((pot(i) / 128 - 1) * 5, "#0.0##")
    datos(i, 13) = Format(-(offset(i) / 128 - 1), "#0.0##")
    datos(i, 10) = ""
  End If


  If acttip(i) = 0 Then 'sin función de activación
    datos(i, 14) = "no"
    datos(i, 15) = ""
    datos(i, 16) = ""
    datos(i, 17) = ""
    datos(i, 18) = ""
  End If
  If acttip(i) = 1 Then 'Escalón
    datos(i, 14) = "step"
    datos(i, 15) = Format(umbral(i), "#0.0##")
    datos(i, 16) = ""
    datos(i, 17) = ""
    datos(i, 18) = ""
  End If
  If acttip(i) = 2 Then 'Rampa
    datos(i, 14) = "ramp"
    datos(i, 15) = ""
    datos(i, 16) = Format(pend(i), "#0.0##")
    datos(i, 17) = Format((sat(i) / 256) * 5, "#0.0##")
    datos(i, 18) = Format((corr(i) / 128 - 1) * 2.5, "#0.0##")
  End If
  If acttip(i) = 3 Then 'Sigmoidal
    datos(i, 14) = "tanh"
    datos(i, 15) = ""
    datos(i, 16) = Format((pends(i) / 256) * 5, "#0.0##")
    datos(i, 17) = Format((sats(i) / 256) * 4, "#0.0##")
    datos(i, 18) = Format(-2.5 * (corrs(i) / 128 - 1), "#0.0##")
  End If
  datos(i, 19) = Format((pulso(i) / 256) * 25, "#0.0##")
  datos(i, 20) = Format((retardo(i) / 256) * 50, "#0.0##")
Next i

For i = 1 To 10
  For j = 1 To 20
    Grid1.Col = j
    Grid1.Row = i
    Grid1.Text = datos(i, j)
  Next j
Next i

End Sub



Private Sub mnuAbrir_Click()
  Dim pasobyte(230) As Byte, paso As Byte
  Dim i As Integer, j As Integer
  OpenFileName = ""
  CommonDialog1.filename = ""
  CommonDialog1.DefaultExt = "*.dat"
  CommonDialog1.Filter = "Archivos de datos
(*.dat)|*.dat|Todos los archivos (*.*)|*.*"
  CommonDialog1.ShowOpen
  If Err <> 32755 Then   ' User chose Cancel.
    OpenFileName = CommonDialog1.filename
    ' If the file is larger than 65K, it can't
    ' be opened, so cancel the operation.
    If FileLen(OpenFileName) > 65000 Then
      MsgBox "El archivo es muy grande."
      Exit Sub
    End If
  End If
  fhandler = FreeFile
  Open OpenFileName For Input As #fhandler
  i = 1
  Do While Not EOF(fhandler)
    Input #fhandler, paso
    pasobyte(i) = paso
    Debug.Print pasobyte(i), i
    i = i + 1
  Loop
  Close #fhandler
  'Pasa los datos a los arreglos de memoria
  'Este orden será el establecido: cada parámetro y para cada
una de las diez neuronas
  'peso(10,8) son los primeros ochenta
  For i = 1 To 10
    For j = 1 To 8
      peso(i, j) = pasobyte(((j - 1) * 10) + i)
    Next
    inttip(i) = pasobyte(80 + i)
    disparo(i) = pasobyte(90 + i)
    tau(i) = pasobyte(100 + i)
    pot(i) = pasobyte(110 + i)
    offset(i) = pasobyte(120 + i)
    acttip(i) = pasobyte(130 + i)
    umbral(i) = pasobyte(140 + i)
    pend(i) = pasobyte(150 + i)
    sat(i) = pasobyte(160 + i)
    corr(i) = pasobyte(170 + i)
    pends(i) = pasobyte(180 + i)
    sats(i) = pasobyte(190 + i)
    corrs(i) = pasobyte(200 + i)
    pulso(i) = pasobyte(210 + i)
    retardo(i) = pasobyte(220 + i)
  Next
  'Ahora se manda a escribir al grid
  llenagrid
'  ReDim pasobyte(0)
End Sub

Private Sub mnuescalon_Click()
  frmactiv.Show
End Sub


Private Sub mnuGuardar_Click()
On Error GoTo ErrHandler
  Dim pasobyte(230) As Byte, paso As Byte
  Dim i As Integer, j As Integer
  'Subir los datos al arreglo de pasobyte
  For i = 1 To 10
    For j = 1 To 8
      pasobyte(((j - 1) * 10) + i) = peso(i, j)
    Next
    pasobyte(80 + i) = inttip(i)
    pasobyte(90 + i) = disparo(i)
    pasobyte(100 + i) = tau(i)
    pasobyte(110 + i) = pot(i)
    pasobyte(120 + i) = offset(i)
    pasobyte(130 + i) = acttip(i)
    pasobyte(140 + i) = umbral(i)
    pasobyte(150 + i) = pend(i)
    pasobyte(160 + i) = sat(i)
    pasobyte(170 + i) = corr(i)
    pasobyte(180 + i) = pends(i)
```

```
    pasobyte(190 + i) = sats(i)
    pasobyte(200 + i) = corrs(i)
    pasobyte(210 + i) = pulso(i)
    pasobyte(220 + i) = retardo(i)
   Next
  'Pide el nombre del archivo
  CloseFileName = OpenFileName
  CommonDialog1.filename = CloseFileName
  CommonDialog1.DefaultExt = "*.dat"
  CommonDialog1.Filter = "Archivos de datos
(*.dat)|*.dat|Todos los archivos (*.*)|*.*"
  CommonDialog1.InitDir = cdlOFNNoChangeDir
  CommonDialog1.ShowSave
  If (CommonDialog1.filename <> "") Then    ' User chose
Cancel.
    CloseFileName = CommonDialog1.filename
    fhandler = FreeFile
    Open CloseFileName For Output Shared As #fhandler
    For i = 1 To 230
      Print #fhandler, pasobyte(i)
    Next
    Close fhandler
  End If
  OpenFileName = CloseFileName
  Exit Sub
EnError:
  Exit Sub
End Sub

Private Sub mnuGuardarcomo_Click()
  mnuGuardar_Click
End Sub


Private Sub mnuneu1_Click()
    neur = 1
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu10_Click()
    neur = 10
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu2_Click()
    neur = 2
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu3_Click()
    neur = 3
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu4_Click()
    neur = 4
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu5_Click()
    neur = 5
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu6_Click()
    neur = 6
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu7_Click()
    neur = 7
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu8_Click()
    neur = 8
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnuneu9_Click()
    neur = 9
    VScroll1.Value = neur
    Label1.Caption = neur
End Sub

Private Sub mnurampa_Click()
    frmactiv.Show
End Sub

Private Sub mnusigmoide_Click()
    frmactiv.Show

End Sub

Private Sub mnuSalir_Click()
  Close
  End
End Sub

Private Sub SSCommand1_MouseMove(Button As Integer,
Shift As Integer, X As Single, Y As Single)
    intbtnselected = 1
    Timer1.Enabled = True

End Sub


Private Sub SSCommand2_MouseMove(Button As Integer,
Shift As Integer, X As Single, Y As Single)
    intbtnselected = 2
    Timer1.Enabled = True

End Sub

Private Sub SSCommand3_Click()
    neur = Val(VScroll1.Value)
    frmactiv.Show
End Sub

Private Sub SSCommand3_MouseMove(Button As Integer,
Shift As Integer, X As Single, Y As Single)
    intbtnselected = 3
    Timer1.Enabled = True
```

```
End Sub

Private Sub picToolBar_MouseMove(Button As Integer, Shift
As Integer, X As Single, Y As Single)
  txtToolTip.Visible = False
  Timer1.Enabled = False
End Sub

Sub ShowTip()
  Select Case intbtnselected
  ' Despliega el ToolTip para el boton de la barra
  Case Is = 1
    txtToolTip.Top = SSCommand1.Top * 2 + 300
    txtToolTip.Left = SSCommand1.Left + 200
    txtToolTip.Text = "Adder"
    txtToolTip.Visible = True
  Case Is = 2
    txtToolTip.Top = SSCommand2.Top * 2 + 300
    txtToolTip.Left = SSCommand2.Left + 200
    txtToolTip.Text = "Integrator"
    txtToolTip.Visible = True
  Case Is = 3
    txtToolTip.Top = SSCommand3.Top * 2 + 300
    txtToolTip.Left = SSCommand3.Left + 200
    txtToolTip.Text = "Step"
    txtToolTip.Visible = True
  Case Is = 4
    txtToolTip.Top = SSCommand4.Top * 2 + 300
    txtToolTip.Left = SSCommand4.Left + 200
    txtToolTip.Text = "Ramp"
    txtToolTip.Visible = True
  Case Is = 5
    txtToolTip.Top = SSCommand5.Top * 2 + 300
    txtToolTip.Left = SSCommand5.Left + 200
    txtToolTip.Text = "Tanh"
    txtToolTip.Visible = True
  Case Is = 6
    txtToolTip.Top = SSCommand6.Top * 2 + 300
    txtToolTip.Left = SSCommand6.Left + 200
    txtToolTip.Text = "Pulse"
    txtToolTip.Visible = True

  End Select

End Sub


Private Sub mnuintegra_Click()
  frmintegra.Show
End Sub

Private Sub mnupulso_Click()
  frmretardo.Show
End Sub


Private Sub mnusuma_Click()
  frmsuma.Show
End Sub


Private Sub SSCommand1_Click()
  frmsuma.Show
End Sub


Private Sub SSCommand2_Click()
  frmintegra.Show
End Sub


Private Sub SSCommand4_Click()
  neur = Val(VScroll1.Value)
  frmactiv.Show
End Sub

Private Sub SSCommand4_MouseMove(Button As Integer,
Shift As Integer, X As Single, Y As Single)
  intbtnselected = 4
  Timer1.Enabled = True
End Sub


Private Sub SSCommand5_Click()
  neur = Val(VScroll1.Value)
  frmactiv.Show
End Sub

Private Sub SSCommand5_MouseMove(Button As Integer,
Shift As Integer, X As Single, Y As Single)
  intbtnselected = 5
  Timer1.Enabled = True
End Sub


Private Sub SSCommand6_Click()
  frmretardo.Show
End Sub


Private Sub SSCommand6_MouseMove(Button As Integer,
Shift As Integer, X As Single, Y As Single)
  intbtnselected = 6
  Timer1.Enabled = True
End Sub


Private Sub SSCommand7_Click()
  SSCommand7.Enabled = False


  comm1.CommPort = 2
  ' 2400 baud, no parity, 8 data, and 1 stop bit.
  comm1.Settings = "2400,N,8,1"
  ' Tell the control to read entire buffer when Input is used.
  comm1.InputLen = 0
  ' Open the port.


  comm1.PortOpen = True

' Envia una i
  comm1.Output = "i"
  ' Espera a recibir un dato
  Do
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input
  'Envia el comando
  comm1.Output = Chr(0) 'selecciona primer comando
  ' Espera a recibir un dato (una i)
  Do
```

```
   Dummy = DoEvents()
Loop Until comm1.InBufferCount >= 1
' Lee el dato del puerto serial
InString$ = comm1.Input


' MANDA LOS DATOS

For i = 1 To 10
  'PESOS
  For j = 1 To 8  'envia los 8 pesos
    comm1.Output = Chr(peso(i, j))

    Do    ' Espera a recibir un dato
      Dummy = DoEvents()
    Loop Until comm1.InBufferCount >= 1
    ' Lee el dato del puerto serial
    InString$ = comm1.Input
  Next j

  'TIPO INTEGRADOR
  comm1.Output = Chr(inttip(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'TIPO DE DISPARO
  comm1.Output = Chr(dispar(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'CONSTANTE DE TIEMPO
  comm1.Output = Chr(tau(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'POTENCIAL DE REPOSO
  comm1.Output = Chr(pot(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'OFFSET DEL INTEGRADOR
  comm1.Output = Chr(offset(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'TIPO DE FUNCION DE ACTIVACION
  comm1.Output = Chr(actfun(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'UMBRAL
  comm1.Output = Chr(umbral(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'PENDIENTE DE LA RAMPA
  comm1.Output = Chr(pend(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'SATURACION DE LA RAMPA
  comm1.Output = Chr(sat(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'CORRIMIENTO DE LA RAMPA
  comm1.Output = Chr(corr(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'PENDIENTE DE LA TANGENTE HIPERBOLICA
  comm1.Output = Chr(pends(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'SATURACION DE LA TANH
  comm1.Output = Chr(sats(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'CORRIMIENTO DE LA TANH
  comm1.Output = Chr(corrs(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input


  'DURACION DEL PULSO
  comm1.Output = Chr(pulso(i))
  Do    ' Espera a recibir un dato
    Dummy = DoEvents()
  Loop Until comm1.InBufferCount >= 1
  ' Lee el dato del puerto serial
  InString$ = comm1.Input

  'DURACION DEL RETARDO
```

```
    comm1.Output = Chr(retardo(i))
    Do    ' Espera a recibir un dato
      Dummy = DoEvents()
    Loop Until comm1.InBufferCount >= 1
    ' Lee el dato del puerto serial
    InString$ = comm1.Input


  Next i

  ' Cierra el puerto serie
  comm1.PortOpen = False

  SSCommand7.Enabled = True

End Sub

Private Sub Timer1_Timer()
  ShowTip    'Llama al procedimiento ShowTip
End Sub


Private Sub VScroll1_Change()
  neur = Val(VScroll1.Value)
  Label1.Caption = neur
End Sub


Attribute VB_Name = "Module1"
Global neur As Single 'indice del numero de neurona
Global peso(10, 8) As Byte 'peso(neur, num de entrada)
Global inttip(10) As Byte
Global disparo(10) As Byte 'tipo de disparo del integrador
Global tau(10) As Byte 'Constante de tiempo del integrador
Global pot(10) As Byte 'Potencial de reposo del integrador
Global offset(10) As Byte 'ajuste de offset del integrador
Global acttip(10) As Byte 'tipo de función de activación
Global umbral(10) As Byte 'Umbral del escalón
Global pend(10) As Byte 'Pendiente de la rampa
Global sat(10) As Byte 'Saturación de la rampa
Global corr(10) As Byte 'Corrimiento de la rampa
Global pends(10) As Byte 'Pendiente de la sigmoide
Global sats(10) As Byte 'Saturación de la sigmoide
Global corrs(10) As Byte 'Corrimiento de la sigmoide
Global pulso(10) As Byte 'Duración del pulso
Global retardo(10) As Byte 'Duración del retardo
Global i, j As Integer 'indice
Global datos(10, 20) As String

'Para manejo de archivos
Public OpenFileName As String, CloseFileName As String
Public fhandler As Byte

VERSION 4.00
Begin VB.Form frmsuma

Attribute VB_Name = "frmsuma"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
  Dim dato1, dato2, dato3, dato4 As Single
  Dim dato5, dato6, dato7, dato8 As Single
  Dim buff1, buff2, buff3, buff4 As Byte
  Dim buff5, buff6, buff7, buff8 As Byte
Private Sub cmd1_Click()
  peso(neur, 1) = Val(HScroll1.Value)
  peso(neur, 2) = Val(HScroll2.Value)
  peso(neur, 3) = Val(HScroll3.Value)
  peso(neur, 4) = Val(HScroll4.Value)
  peso(neur, 5) = Val(HScroll5.Value)
  peso(neur, 6) = Val(HScroll6.Value)
  peso(neur, 7) = Val(HScroll7.Value)
  peso(neur, 8) = Val(HScroll8.Value)

  Call MDIForm1.llenagrid

  Unload frmsuma

End Sub


Private Sub cmd2_Click()
  peso(neur, 1) = buff1
  peso(neur, 2) = buff2
  peso(neur, 3) = buff3
  peso(neur, 4) = buff4
  peso(neur, 5) = buff5
  peso(neur, 6) = buff6
  peso(neur, 7) = buff7
  peso(neur, 8) = buff8

  Unload frmsuma
End Sub


Private Sub Form_Load()

  buff1 = peso(neur, 1)
  buff2 = peso(neur, 2)
  buff3 = peso(neur, 3)
  buff4 = peso(neur, 4)
  buff5 = peso(neur, 5)
  buff6 = peso(neur, 6)
  buff7 = peso(neur, 7)
  buff8 = peso(neur, 8)


  HScroll1.Value = peso(neur, 1)
  dato1 = -(peso(neur, 1) / 128 - 1)
  lbl1.Caption = Format(dato1, "#0.0##")

  HScroll2.Value = peso(neur, 2)
  dato2 = -(peso(neur, 2) / 128 - 1)
  lbl2.Caption = Format(dato2, "#0.0##")

  HScroll3.Value = peso(neur, 3)
  dato3 = -(peso(neur, 3) / 128 - 1)
  lbl3.Caption = Format(dato3, "#0.0##")

  HScroll4.Value = peso(neur, 4)
  dato4 = -(peso(neur, 4) / 128 - 1)
  lbl4.Caption = Format(dato4, "#0.0##")

  HScroll5.Value = peso(neur, 5)
  dato5 = -(peso(neur, 5) / 128 - 1)
  lbl5.Caption = Format(dato5, "#0.0##")

  HScroll6.Value = peso(neur, 6)
  dato6 = -(peso(neur, 6) / 128 - 1)
  lbl6.Caption = Format(dato6, "#0.0##")

  HScroll7.Value = peso(neur, 7)
  dato7 = -(peso(neur, 7) / 128 - 1)
  lbl7.Caption = Format(dato7, "#0.0##")
```

```
  HScroll8.Value = peso(neur, 8)
  dato8 = -(peso(neur, 8) / 128 - 1)
  lbl8.Caption = Format(dato8, "#0.0##")

End Sub

Private Sub HScroll1_Change()
  peso(neur, 1) = Val(HScroll1.Value)
  dato1 = -((peso(neur, 1)) / 128 - 1)
  lbl1.Caption = Format(dato1, "#0.0##")

End Sub


Private Sub HScroll2_Change()
  peso(neur, 2) = Val(HScroll2.Value)
  dato2 = -(peso(neur, 2) / 128 - 1)
  lbl2.Caption = Format(dato2, "#0.0##")

End Sub


Private Sub HScroll3_Change()
  peso(neur, 3) = Val(HScroll3.Value)
  dato3 = -(peso(neur, 3) / 128 - 1)
  lbl3.Caption = Format(dato3, "#0.0##")

End Sub


Private Sub HScroll4_Change()
  peso(neur, 4) = Val(HScroll4.Value)
  dato4 = -(peso(neur, 4) / 128 - 1)
  lbl4.Caption = Format(dato4, "#0.0##")

End Sub


Private Sub HScroll5_Change()
  peso(neur, 5) = Val(HScroll5.Value)
  dato5 = -(peso(neur, 5) / 128 - 1)
  lbl5.Caption = Format(dato5, "#0.0##")
End Sub


Private Sub HScroll6_Change()
  peso(neur, 6) = Val(HScroll6.Value)
  dato6 = -(peso(neur, 6) / 128 - 1)
  lbl6.Caption = Format(dato6, "#0.0##")
End Sub


Private Sub HScroll7_Change()
  peso(neur, 7) = Val(HScroll7.Value)
  dato7 = -(peso(neur, 7) / 128 - 1)
  lbl7.Caption = Format(dato7, "#0.0##")
End Sub


Private Sub HScroll8_Change()
  peso(neur, 8) = Val(HScroll8.Value)
  dato8 = -(peso(neur, 8) / 128 - 1)
  lbl8.Caption = Format(dato8, "#0.0##")
End Sub


VERSION 4.00
```

```
Begin VB.Form frmintegra

Attribute VB_Name = "frmintegra"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
  Dim buff1, buff2, buff3, buff4 As Byte
  Dim dato1, dato2, dato3, dato4 As Single


Private Sub Command1_Click()

  If opt1.Value = True Then inttip(neur) = 0 'ninguno
  If opt2.Value = True Then inttip(neur) = 1 'disparo
  If opt3.Value = True Then inttip(neur) = 2 'fugas

  tau(neur) = Val(HScroll1.Value)
  pot(neur) = Val(HScroll2.Value)
  offset(neur) = Val(HScroll3.Value)

  Call MDIForm1.llenagrid

  Unload frmintegra

End Sub

Private Sub Command2_Click()

  tau(neur) = buff1
  pot(neur) = buff2
  offset(neur) = buff3

  Unload frmintegra

End Sub


Private Sub Form_Load()

  opt1.Value = False
  opt2.Value = False
  opt3.Value = False
  Frame2.Enabled = False
  opt4.Enabled = False
  opt5.Enabled = False



  If inttip(neur) = 0 Then opt1.Value = True 'ninguno
  If inttip(neur) = 1 Then
  opt2.Value = True 'disparo
  Frame2.Enabled = True
  opt4.Enabled = True
  opt5.Enabled = True
  End If
  If inttip(neur) = 2 Then opt3.Value = True 'fugas

  buff1 = tau(neur)
  buff2 = pot(neur)
  buff3 = offset(neur)

  HScroll1.Value = tau(neur)
  dato1 = (tau(neur) / 255)
  lbl1.Caption = Format(dato1, "#0.0##")

  HScroll2.Value = pot(neur)
  dato2 = (pot(neur) / 128 - 1) * 5
  lbl2.Caption = Format(dato2, "#0.0##")
```

```
  HScroll3.Value = offset(neur)
  dato3 = -(offset(neur) / 128 - 1)
  lbl3.Caption = Format(dato3, "#0.0##")



End Sub


Private Sub HScroll1_Change()
  tau(neur) = Val(HScroll1.Value)
  dato1 = (tau(neur) / 255)
  lbl1.Caption = Format(dato1, "#0.0##")

End Sub



Private Sub HScroll2_Change()
  pot(neur) = Val(HScroll2.Value)
  dato2 = (pot(neur) / 128 - 1) * 5
  lbl2.Caption = Format(dato2, "#0.0##")

End Sub


Private Sub HScroll3_Change()
  offset(neur) = Val(HScroll3.Value)
  dato3 = -(offset(neur) / 128 - 1)
  lbl3.Caption = Format(dato3, "#0.0##")
End Sub


Private Sub opt1_Click()

  inttip(neur) = 0
  opt1.Value = True 'Ninguno
  opt2.Value = False
  opt3.Value = False
  Frame2.Enabled = False
  opt4.Enabled = False
  opt5.Enabled = False
End Sub


Private Sub opt2_Click()

  inttip(neur) = 1
  opt1.Value = False
  opt2.Value = True 'Disparo
  opt3.Value = False
  Frame2.Enabled = True
  opt4.Enabled = True
  opt5.Enabled = True
  If disparo(neur) = 0 Then opt4.Value = True
  If disparo(neur) = 1 Then opt5.Value = True

End Sub


Private Sub opt3_Click()

  inttip(neur) = 2
  opt1.Value = False
  opt2.Value = False
  opt3.Value = True 'Fugas
  Frame2.Enabled = False
  opt4.Enabled = False
  opt5.Enabled = False
End Sub


Private Sub opt4_Click()
  disparo(neur) = 0 'Escalón
End Sub


Private Sub opt5_Click()
  disparo(neur) = 1
End Sub

VERSION 4.00
Begin VB.Form frmactiv

End
Attribute VB_Name = "frmactiv"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Dim dato1, dato2, dato3, dato4, dato5 As Single
Dim dato6, dato7 As Single
Dim buff1, buff2, buff3, buff4, buff5 As Single
Dim buff6, buff7 As Single

Private Sub Command1_Click()
  umbral(neur) = Val(HScroll1.Value)
  pend(neur) = Val(HScroll2.Value)
  sat(neur) = Val(HScroll3.Value)
  corr(neur) = Val(HScroll4.Value)
  pends(neur) = Val(HScroll5.Value)
  sats(neur) = Val(HScroll6.Value)
  corrs(neur) = Val(HScroll7.Value)

  If Option1 = True Then acttip(neur) = 0  'ninguna
  If Option2 = True Then acttip(neur) = 1  'escalón
  If Option3 = True Then acttip(neur) = 2  'rampa
  If Option4 = True Then acttip(neur) = 3  'sigmoidal

  Call MDIForm1.llenagrid

  Unload frmactiv

End Sub

Private Sub Command2_Click()
  umbral(neur) = buff1
  pend(neur) = buff2
  sat(neur) = buff3
  corr(neur) = buff4
  pends(neur) = buff5
  sats(neur) = buff6
  corrs(neur) = buff7

  Unload frmactiv

End Sub

Private Sub Form_Load()
  buff1 = umbral(neur)
  buff2 = pend(neur)
```

```
buff3 = sat(neur)
buff4 = corr(neur)
buff5 = pends(neur)
buff6 = sats(neur)
buff7 = corrs(neur)


Label8.Caption = neur

HScroll1.Value = umbral(neur)
dato1 = (umbral(neur) / 128 - 1) * 5
lbl1.Caption = Format(dato1, "#0.0##")

HScroll2.Value = pend(neur)
dato2 = pend(neur)
lbl2.Caption = Format(dato2, "#0.0##")

HScroll3.Value = sat(neur)
dato3 = (sat(neur) / 256) * 5
lbl3.Caption = Format(dato3, "#0.0##")

HScroll4.Value = corr(neur)
dato4 = (corr(neur) / 128 - 1) * 2.5
lbl4.Caption = Format(dato4, "#0.0##")

HScroll5.Value = pends(neur)
dato5 = (pends(neur) / 256) * 5
lbl5.Caption = Format(dato5, "#0.0##")

HScroll6.Value = sats(neur)
dato6 = (sats(neur) / 256) * 4
lbl6.Caption = Format(dato6, "#0.0##")

HScroll7.Value = corrs(neur)
dato7 = -2.5 * (corrs(neur) / 128 - 1)
lbl7.Caption = Format(dato7, "#0.0##")

Option1.Value = False
Option2.Value = False
Option3.Value = False
Option4.Value = False


If acttip(neur) = 0 Then 'ninguna
   Option1.Value = True
   Frame1.Enabled = False
   HScroll1.Enabled = False
   Label1.Enabled = False
   Frame2.Enabled = False
   HScroll2.Enabled = False
   HScroll3.Enabled = False
   HScroll4.Enabled = False
   Label2.Enabled = False
   Label3.Enabled = False
   Label4.Enabled = False
   Frame3.Enabled = False
   HScroll5.Enabled = False
   HScroll6.Enabled = False
   HScroll7.Enabled = False
   Label5.Enabled = False
   Label6.Enabled = False
   Label7.Enabled = False
   lbl1.Enabled = False
   lbl2.Enabled = False
   lbl3.Enabled = False
   lbl4.Enabled = False
   lbl5.Enabled = False
   lbl6.Enabled = False
   lbl7.Enabled = False


End If
If acttip(neur) = 1 Then 'escalón
   Option2.Value = True
   Frame1.Enabled = True
   HScroll1.Enabled = True
   Label1.Enabled = True
   Frame2.Enabled = False
   HScroll2.Enabled = False
   HScroll3.Enabled = False
   HScroll4.Enabled = False
   Label2.Enabled = False
   Label3.Enabled = False
   Label4.Enabled = False
   Frame3.Enabled = False
   HScroll5.Enabled = False
   HScroll6.Enabled = False
   HScroll7.Enabled = False
   Label5.Enabled = False
   Label6.Enabled = False
   Label7.Enabled = False
   lbl1.Enabled = True
   lbl2.Enabled = False
   lbl3.Enabled = False
   lbl4.Enabled = False
   lbl5.Enabled = False
   lbl6.Enabled = False
   lbl7.Enabled = False
End If

If acttip(neur) = 2 Then 'rampa
   Option3.Value = True
   Frame1.Enabled = False
   HScroll1.Enabled = False
   Label1.Enabled = False
   Frame2.Enabled = True
   HScroll2.Enabled = True
   HScroll3.Enabled = True
   HScroll4.Enabled = True
   Label2.Enabled = True
   Label3.Enabled = True
   Label4.Enabled = True
   Frame3.Enabled = False
   HScroll5.Enabled = False
   HScroll6.Enabled = False
   HScroll7.Enabled = False
   Label5.Enabled = False
   Label6.Enabled = False
   Label7.Enabled = False
   lbl1.Enabled = False
   lbl2.Enabled = True
   lbl3.Enabled = True
   lbl4.Enabled = True
   lbl5.Enabled = False
   lbl6.Enabled = False
   lbl7.Enabled = False
End If

If acttip(neur) = 3 Then 'sigmoidal
   Option4.Value = True
   Frame1.Enabled = False
   HScroll1.Enabled = False
   Label1.Enabled = False
   Frame2.Enabled = False
```

```
    HScroll2.Enabled = False
    HScroll3.Enabled = False
    HScroll4.Enabled = False
    Label2.Enabled = False
    Label3.Enabled = False
    Label4.Enabled = False
    Frame3.Enabled = True
    HScroll5.Enabled = True
    HScroll6.Enabled = True
    HScroll7.Enabled = True
    Label5.Enabled = True
    Label6.Enabled = True
    Label7.Enabled = True
    lbl1.Enabled = False
    lbl2.Enabled = False
    lbl3.Enabled = False
    lbl4.Enabled = False
    lbl5.Enabled = True
    lbl6.Enabled = True
    lbl7.Enabled = True

  End If


End Sub

Private Sub HScroll1_Change()
  umbral(neur) = Val(HScroll1.Value)
  dato1 = (umbral(neur) / 128 - 1) * 5
  lbl1.Caption = Format(dato1, "#0.0##")
End Sub

Private Sub HScroll2_Change()
  pend(neur) = Val(HScroll2.Value)
  dato2 = pend(neur)
  lbl2.Caption = Format(dato2, "#0.0##")
End Sub

Private Sub HScroll3_Change()
  sat(neur) = Val(HScroll3.Value)
  dato3 = (sat(neur) / 256) * 5
  lbl3.Caption = Format(dato3, "#0.0##")
End Sub

Private Sub HScroll4_Change()
  corr(neur) = Val(HScroll4.Value)
  dato4 = (corr(neur) / 128 - 1) * 2.5
  lbl4.Caption = Format(dato4, "#0.0##")
End Sub

Private Sub HScroll5_Change()
  pends(neur) = Val(HScroll5.Value)
  dato5 = (pends(neur) / 256) * 5
  lbl5.Caption = Format(dato5, "#0.0##")
End Sub

Private Sub HScroll6_Change()
  sats(neur) = Val(HScroll6.Value)
  dato6 = (sats(neur) / 256) * 4
  lbl6.Caption = Format(dato6, "#0.0##")
End Sub

Private Sub HScroll7_Change()
  corrs(neur) = Val(HScroll7.Value)
  dato7 = -2.5 * (corrs(neur) / 128 - 1)
  lbl7.Caption = Format(dato7, "#0.0##")
End Sub
```

```
Private Sub Option1_Click() 'ninguna

  acttip(neur) = 0
    Frame1.Enabled = False
    HScroll1.Enabled = False
    Label1.Enabled = False
    Frame2.Enabled = False
    HScroll2.Enabled = False
    HScroll3.Enabled = False
    HScroll4.Enabled = False
    Label2.Enabled = False
    Label3.Enabled = False
    Label4.Enabled = False
    Frame3.Enabled = False
    HScroll5.Enabled = False
    HScroll6.Enabled = False
    HScroll7.Enabled = False
    Label5.Enabled = False
    Label6.Enabled = False
    Label7.Enabled = False
    lbl1.Enabled = False
    lbl2.Enabled = False
    lbl3.Enabled = False
    lbl4.Enabled = False
    lbl5.Enabled = False
    lbl6.Enabled    False
    lbl7.Enabled = False

End Sub


Private Sub Option2_Click() 'Escalón

  acttip(neur) = 1
    Frame1.Enabled = True
    HScroll1.Enabled = True
    Label1.Enabled = True
    Frame2.Enabled = False
    HScroll2.Enabled = False
    HScroll3.Enabled = False
    HScroll4.Enabled = False
    Label2.Enabled = False
    Label3.Enabled = False
    Label4.Enabled = False
    Frame3.Enabled = False
    HScroll5.Enabled = False
    HScroll6.Enabled = False
    HScroll7.Enabled = False
    Label5.Enabled = False
    Label6.Enabled = False
    Label7.Enabled = False
    lbl1.Enabled = True
    lbl2.Enabled = False
    lbl3.Enabled = False
    lbl4.Enabled = False
    lbl5.Enabled = False
    lbl6.Enabled = False
    lbl7.Enabled = False
End Sub


Private Sub Option3_Click() 'Rampa

  acttip(neur) = 2
    Frame1.Enabled = False
    HScroll1.Enabled = False
```

```
    Label1.Enabled = False
    Frame2.Enabled = True
    HScroll2.Enabled = True
    HScroll3.Enabled = True
    HScroll4.Enabled = True
    Label2.Enabled = True
    Label3.Enabled = True
    Label4.Enabled = True
    Frame3.Enabled = False
    HScroll5.Enabled = False
    HScroll6.Enabled = False
    HScroll7.Enabled = False
    Label5.Enabled = False
    Label6.Enabled = False
    Label7.Enabled = False
    lbl1.Enabled = False
    lbl2.Enabled = True
    lbl3.Enabled = True
    lbl4.Enabled = True
    lbl5.Enabled = False
    lbl6.Enabled = False
    lbl7.Enabled = False


End Sub


Private Sub Option4_Click() 'Sigmoidal

  acttip(neur) = 3
    Frame1.Enabled = False
    HScroll1.Enabled = False
    Label1.Enabled = False
    Frame2.Enabled = False
    HScroll2.Enabled = False
    HScroll3.Enabled = False
    HScroll4.Enabled = False
    Label2.Enabled = False
    Label3.Enabled = False
    Label4.Enabled = False
    Frame3.Enabled = True
    HScroll5.Enabled = True
    HScroll6.Enabled = True
    HScroll7.Enabled = True
    Label5.Enabled = True
    Label6.Enabled = True
    Label7.Enabled = True
    lbl1.Enabled = False
    lbl2.Enabled = False
    lbl3.Enabled = False
    lbl4.Enabled = False
    lbl5.Enabled = True
    lbl6.Enabled = True
    lbl7.Enabled = True

End Sub

VERSION 4.00
Begin VB.Form frmretardo

End
Attribute VB_Name = "frmretardo"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Dim dato1, dato2 As Single
Dim buff1, buff2 As Byte


Private Sub cmd1_Click()

pulso(neur) = Val(HScroll1.Value)
retardo(neur) = Val(HScroll2.Value)

Call MDIForm1.llenagrid

Unload frmretardo


End Sub


Private Sub cmd2_Click()

  pulso(neur) = buff1
  retardo(neur) = buff2

  Unload frmretardo

End Sub

Private Sub Form_Load()
  buff1 = pulso(neur)
  buff2 = retardo(neur)

  HScroll1.Value = pulso(neur)
  dato1 = (pulso(neur) / 256) * 25
  lbl1.Caption = Format(dato1, "#0.0##")

  HScroll2.Value = retardo(neur)
  dato2 = (retardo(neur) / 256) * 50
  lbl2.Caption = Format(dato2, "#0.0##")


End Sub


Private Sub HScroll1_Change()
  pulso(neur) = Val(HScroll1.Value)
  dato1 = (pulso(neur) / 256) * 25
  lbl1.Caption = Format(dato1, "#0.0##")

End Sub

Private Sub HScroll2_Change()
  retardo(neur) = Val(HScroll2.Value)
  dato2 = (retardo(neur) / 256) * 50
  lbl2.Caption = Format(dato2, "#0.0##")

End Sub
```