UNIVERSIDAD DE SEVILLA
Departamento de Electrónica y Electromagnetismo

CATEGORIZADORES NEURONALES EN VLSI

Memoria presentada por
TERESA SERRANO GOTARREDONA
para optar al grado de Doctor en Ciencias Físicas

Sevilla, Julio de 1996

UNIVERSIDAD DE SEVILLA
Departamento de Electrónica y Electromagnetismo

CATEGORIZADORES NEURONALES EN VLSI

Memoria presentada por
TERESA SERRANO GOTARREDONA
para optar al grado de Doctor en Ciencias Físicas

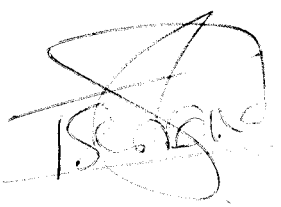
Sevilla, Julio de 1996

CATEGORIZADORES NEURONALES EN VLSI

Memoria presentada por

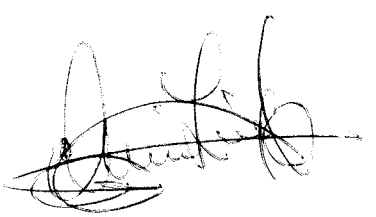
TERESA SERRANO GOTARREDONA

para optar grado de Doctor en Ciencias Físicas



Sevilla, Julio de 1996

EL DIRECTOR:



Bernabé Linares Barranco

Departamento de Electrónica y Electromagnetismo
Universidad de Sevilla

A Bernabé, a mis padres Joaquín y Caridad, y a mis hermanos por su paciencia y cariño.

Contenidos

Nomenclatura

ART	Teoría de la resonancia adaptativa
ART 1	Primer algoritmo de la teoría de la resonancia adaptativa
ARTMAP	Algoritmo de resonancia adaptativa con aprendizaje supervisado
F_1	Capa de las entradas de una arquitectura ART
F_2	Capa de las categorías de una arquitectura ART
N	Número de nudos de la capa F_1
M	Número de nudos de la capa F_2
v_i	Nudo de la capa F_1
u_j	Nudo de la capa F_2
$\mathbf{Z}^{bu}$	Matriz de pesos "bottom-up", que interconectan la capa F_1 con la capa F_2
$\mathbf{z}_j^{bu}$	Vector de pesos que une los nudos de la capa F_1 con del nudo u_j de F_2
z_{ij}^{bu}	Peso que interconecta el nudo v_i de F_1 con el nudo u_j de F_2
$\mathbf{Z}^{td}$	Matriz de pesos "top-down", que interconectan la capa F_2 con la capa F_1
$\mathbf{z}_j^{td}$	Vector de pesos que une el nudo u_j de F_2 con los nudos de la capa F_1
z_{ji}^{td}	Peso que interconecta el nudo u_j de F_2 con el nudo v_i de F_1
ρ	Parámetro de vigilancia
$\boldsymbol{I}$	Patrón de entrada
I_i	Componente del patrón de entrada
$\boldsymbol{X}$	Patrón de activación de la capa F_1
x_i	Estado del nudo v_i
$\boldsymbol{Y}$	Patrón de activación de la capa F_2
y_j	Estado del nudo u_j
$\boldsymbol{S}$	Patrón de señales postsinápsicas de la capa F_1
$h(x_i)$	Señal postsinápsica del nudo v_i de F_1
$\boldsymbol{T}$	Patrón de entrada a la capa F_2
T_j	Señal de entrada al nudo u_j
$\boldsymbol{U}$	Patrón de señales postsinápsicas de la capa F_2
$f(y_j)$	Señal postsinápsica del nudo u_j de F_2
$\boldsymbol{V}$	Patrón "top-down" de entrada a la capa F_1
V_i	Componente de **V** que entra en el nudo u_j.
L	Parámetro del algoritmo ART 1 original
ART 1_m	Algoritmo ART 1 modificado
L_A, L_B, L_M	Parámetros del algoritmo ART 1 modificado
α	Parámetro del algoritmo ART 1 modificado tal que $\alpha = \frac{L_A}{L_B}$
ε	Parámetro del algoritmo ART 1 modificado relacionado con α mediante $\alpha = \frac{1}{1-\varepsilon}$
O_j	Valor inicial de las señal T_j que determina el orden de búsqueda de la categoría u_j
J_i^+	Suma de todas las señales excitatorias que inciden en el nudo v_i
J_i^-	Suma de todas las entradas inhibitorias que actúan en el nudo v_i
J_j^+	Suma de todas las señales excitatorias que inciden en el nudo u_j

J_j^-	Suma de todas las entradas inhibitorias al nudo u_j
s_{ij}	Sinapsis en la columna i y en la fila j en el circuito que implementa el algoritmo ART 1 modificado
N_j	Nudo donde se realiza la suma de corrientes $L_A\sum I_i z_{ij} - L_B\sum z_{ij}$ de la fila j
N_j'	Nudo donde se calcula la suma $L_A\sum I_i z_{ij}$ correspondiente a la fila j
N''	Nudo donde se hace la suma de corrientes $L_A\sum I_i$
T_{Aj}	Valor del término $\sum I_i z_{ij}$ de la fila j
T_{Bj}	Valor del término $\sum z_{ij}$ de la fila j
I_o	Intensidad de salida global del circuito Winner-Take-All en modo de corriente
I_{oj}	Intensidad de salida de la celda j del WTA en modo de corriente
α_j	Parámetro que representa la máxima contribución de la celda j a la salida global en un WTA en modo de corriente
C_C	Capacidad equivalente a la entrada del comparador de corriente de una celda del WTA
G_C	Conductancia equivalente a la entrada del comparador de corriente de una celda del WTA
A	Ganancia en tensión del comparador de corriente de una celda del WTA
v_{xj}	Tensión existente a la entrada del comparador de corriente de una celda del WTA
v_M	Tensión umbral del comparador de corriente
C_A	Capacidad de compensación introducida en cada celda del WTA
C_p	Capacidad equivalente en la "gate" del espejo PMOS del WTA en modo de corriente
C_g	Capacidad parásita "gate-drain" en el transistor que actúa de llave en cada celda del WTA
g_{mn}	Transconductancia del transistor que actúa como llave en una celda del WTA
g_{mp}	Transconductancia en el transistor de entrada del espejo PMOS del WTA
g_n	Conductancia de salida de los transistores de salida del espejo NMOS de cada celda del WTA
P	Parámetro eléctrico de un transistor
ΔP	Desviación del parámetro P entre dos transistores
$P(x, y)$	Función densidad del parámetro eléctrico P
$G(x, y)$	Función de geometría de una configuración de transistores
D_w	Diámetro del "wafer"
Ω	Area del "wafer"
A_P	Parámetro estadístico que modela la contribución del error aleatorio en la desviación del parámetro P
S_P	Parámetro estadístico que modela la contribución del error sistemático en la desviación del parámetro eléctrico P
$r(P_1, P_2)$	Correlación entre los parámetros P_1 y P_2
$\sigma(P_1, P_2)$	Desviación estándar de los valores de $r(P_1, P_2)$ medidos en los distintos chips
n^{max}	Numero de veces que una celda está repetida en la dirección x en el chip para la caracterización del "mismatching"
m^{max}	Número de veces que una celda está repetida en la dirección y en el chip para la caracterización del "mismatching"
$I_o^s(x, y)$	Superficie definida por las corrientes simuladas de salida de un espejo de salida multiple

$I_o^p(x, y)$	Superficie definida por el plano de mejor ajuste de las corrientes de salida de un espejo de salida múltiple
$I_o^m(x, y)$	Superficie definida por las corrientes medidas de salida de un espejo de salida multiple
$I_{maxplane}$	Máximo del plano de mejor ajuste de las corrientes de salida de un espejo de salida múltiple
$I_{minplane}$	Mínimo del plano de mejor ajuste de las corrientes de salida de un espejo de salida múltiple
Inter-ART	Módulo de interconexión de los módulos ART 1 en una arquitectura ARTMAP
$ART\ 1^a$	Módulo ART 1 cuyo parámetro de vigilancia está controlado por el módulo inter-ART en una arquitectura ARTMAP
$ART\ 1^b$	Módulo ART 1 de una arquitectura ARTMAP cuyo parámetro de vigilancia es fijado externamente
a	Vector de entrada al módulo $ART\ 1^a$
b	Vector de entrada al módulo $ART\ 1^b$
F^{ab}	Capa del módulo inter-ART
u_k^{ab}	Nudo k de la capa F^{ab}
y_k^{ab}	Actividad del nudo u_k^{ab}
w_{jk}	Peso que interconecta el nudo u_k^{ab} de la capa F^{ab} con el nudo u_j^a de la capa F_2^a

Categorizadores Neuronales en VLSI

Durante los pasados años ha habido un desarrollo muy fuerte en el campo de las denominadas "Redes Neuronales". Este crecimiento se debe a que los sistemas basados en sus principios son capaces de realizar tareas tales como reconocimiento de voz o de imágenes, clasificación de patrones, memorias asociativas, etc [1]. Tareas fácilmente realizables por seres humanos de forma natural, pero tremendamente complicadas para ser realizadas por ordenadores convencionales. Las redes neuranales ya están siendo utilizadas para realizar tareas que tradicionalmente hubiera sido impensable que fueran realizadas por máquinas [2].

Existen en la literatura diversas propuestas de redes neuronales, con muy diversos mecanismos de aprendizaje y entrenamiento [3]. Todos ellos se han realizado con programas en software que simulan las ecuaciones que definen el comportamiento de las redes neuronales mediante un ordenador convencional. Ésta técnica, aunque suficiente para validar los modelos teóricos de las redes neuronales, resulta en la práctica excesivamente lenta debido a la gran cantidad de cálculos a realizar, consecuencia del tremendo paralelismo de las redes neuronales. Para aplicaciones prácticas es imprescindible disponer de hardware especializado capaz de responder en márgenes de tiempo adecuados.

En el campo de las realizaciones hardware de redes neuronales es posible distinguir dos tipos de implementaciones:

- Las realizaciones hardware de sistemas de "propósito general". Estos son sistemas diseñados para que admitan gran flexibilidad en su topología y en las operaciones de redes neuronales que son capaces de realizar. Estos sistemas son de gran utilidad para los diseñadores de algoritmos de redes neuronales ya que permiten la realización de las operaciones del algoritmo a una velocidad muy superior a la de los ordenadores convencionales.
- Diseño de sistemas para aplicaciones específicas. Para ello es necesario seleccionar previamente el algoritmo de redes neuronales que mejor se adapte al propósito de la aplicación y permita una realización hardware más eficiente.

El presente trabajo de tesis doctoral se ha encaminado a la implementación de un categorizador de patrones binarios en tiempo real. Para nuestra aplicación, hemos elegido el algoritmo ART 1 de categorización de patrones binarios, debido a sus atractivas propiedades orientadas a la realización hardware y a sus propiedades computacionales. Para obtener una realización de circuito más eficiente hemos modificado ligeramente el algoritmo matemático ART 1. Nuestra modificación permite una realización hardware utilizando bloques de circuito más simples, pero conservando todas las propiedades computacionales del algoritmo ART 1 original.

La memoria del trabajo está estructurada de la siguiente manera. Consta de un resumen en castellano de todo el trabajo el cual va referenciando a cinco Apéndices, escritos en inglés, que desarrollan con más detalle los temas que se han cubierto.

En la Sección 1 del presente resumen se explica la operación de un sistema categorizador y las principales características de los algoritmos ART. En el Apéndice 1, cuyo contenido está resumido en la Sección 2 de este

resumen, se describen las modificaciones del algoritmo matemático ART 1 que permiten una realización harware más eficiente. La Sección 3 introduce el circuito que realiza la versión modificada del algoritmo ART 1. Los detalles de la implementación y los resultados experimentales están incluídos en el Apéndice 2. El Apéndice 3 explica la realización de un circuito "Winner-Take-All" en modo de corriente para implementar el proceso de selección de categorías del algoritmo ART 1. En el Apéndice 4, se explica un procedimiento para la caracterización del mismatching de un proceso tecnológico CMOS y se incluyen los resultados de la caracterización de dos tecnologías: la de ES2-1.0μm y la de CNM-2.5μm. Basándonos en los resultados de la caracterización del Apéndice 4 hemos diseñado un nuevo prototipo del chip ART 1 que mantienen la precisión de la operación con un área más reducida. Los resultados experimentales de la operación de este nuevo prototipo están detallados en el Apéndice 5. También en el Apéndice 5, que es una ampliación de la Sección 4 de este capítulo, se incluyen resultados experimentales de sistemas multichip realizados con este nuevo prototipo. Por último, en la Sección 5 se resumen las aportaciones de esta tesis.

1. Sistemas Categorizadores

Un sistema categorizador es aquel que recibe una serie de patrones de entrada y realiza una agrupación de dichos patrones de entrada atendiendo a algún criterio de semejanza.

La operación de un sistema categorizador está ilustrada en la Fig. 1. El sistema categorizador recibe como estímulos externos una secuencia de patrones de entrada **I**, **J**, **K** ... y los va agrupando por semejanza en unas categorías 1, 2, ... Las categorías no existen a priori, sino que se van formando y sus características se van definiendo conforme se reciben los patrones externos o estímulos. Esto contrasta con los llamados "sistemas clasificadores" en los que las diferentes categorías existen a priori (su número y características están predefinidos), y su operación se reduce a decidir qué categoría es más apropiada para clasificar a cada patrón externo que vaya llegando. El categorizador de la Fig. 1 considera semejantes los patrones **I** y **J**, clasificándolos en la categoría 1, mientras que el patrón **K** lo considera diferente y lo clasifica en otra categoría.

Existen sistemas categorizadores que deben ser entrenados "off-line" para construir las categorías a partir de un conjunto de patrones de entrada [4]-[9]. En estos sistemas, el número y características de las categorías son calculadas mediante algún algoritmo previamente a la operación del sistema. Una vez que se ha realizado el entrenamiento, el sistema ya está preparado para operar con el conjunto de patrones para el que ha sido entrenado. A partir de este momento, actúa como un "sistema clasificador". Sin embargo, si se desea

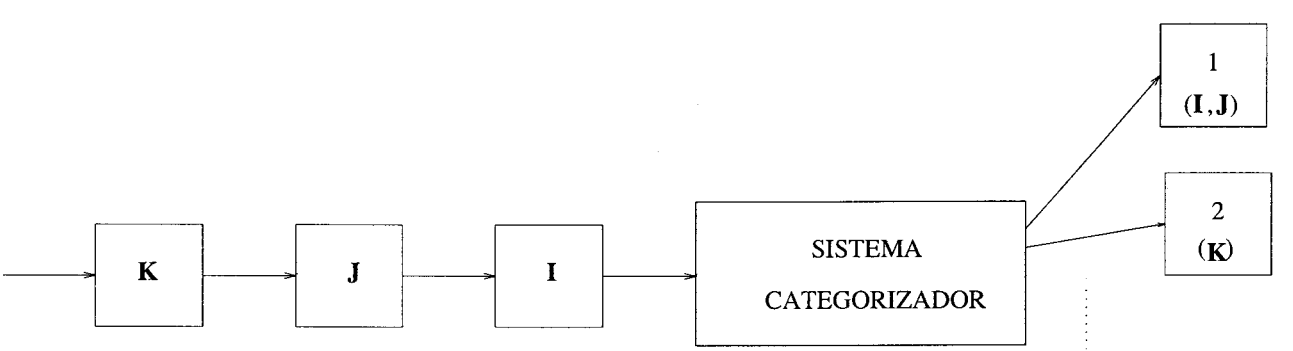


Fig. 1: Esquema de la Operación de un Sistema Categorizador

incorporar algún patrón de entrada nuevo, el sistema debe ser re-entrenado (borrando el conocimiento ya existente) con el nuevo conjunto de patrones de entrada formado por los patrones ya existentes más los nuevos patrones a incorporar.

Por el contrario, se dice que el sistema categorizador opera en "tiempo real" cuando el sistema es capaz de ir formando y adaptando sus categorías internas conforme se le van presentando los patrones de entrada. Cada vez que se presenta un patrón de entrada, el sistema busca la categoría que mayor semejanza guarda con dicho patrón de entrada. A continuación, el sistema actualiza la definición de la categoría seleccionada para incorporar las características relevantes del nuevo patrón. En el caso de que ninguna de las categorías ya existentes sea suficientemente parecida al patrón de entrada presentado, se forma una nueva categoría para clasificar este patrón de entrada.

Las arquitecturas basadas en la Teoría de la Resonancia Adaptativa (ART) constituyen sistemas categorizadores en tiempo real [10]-[15]. Estas arquitecturas son:

- ART 1 [10]: es un categorizador en tiempo real para patrones formados por "pixels" binarios.
- ART 2 [11] y Fuzzy-ART [12]: hacen lo mismo que la arquitectura ART 1 pero con patrones de entrada con pixels analógicos.
- ART 3 [13]: clasifica en tiempo real secuencias de patrones de entrada analógicos y asíncronos (en las demás arquitecturas debe mantenerse una sincronización al presentar los patrones de entrada).
- Las arquitecturas ARTMAP [14] y Fuzzy-ARTMAP [15]: pueden aprender, de forma supervisada, correspondencias entre pares de patrones de entrada binarios y analógicos, respectivamente.

Las arquitecturas ART presentan una serie de propiedades interesantes de las que carecen otros algoritmos categorizadores. Entre estas propiedades destacan:

- *Vigilancia variable*: existe un parámetro ajustable ρ, conocido como parámetro de vigilancia, que ajusta el grado de semejanza que debe existir entre los patrones de entrada para ser clasificados por el sistema en una misma categoría. Un parámetro de vigilancia ρ próximo a '0' hace que el criterio de vigilancia sea poco estricto. El sistema clasificará patrones de entrada bastante diferentes como pertenecientes a una misma categoría. Por el contrario, una vigilancia ρ próxima a '1' hace que se formen categorías más finas, y el número de categorías formadas será mayor.
- *Auto-ajuste del orden de búsqueda*: el orden de búsqueda no está predeterminado sino que se va modificando a medida que el conocimiento del sistema va aumentando.
- *Acceso directo a patrones de entrada familiares*: el sistema siempre accede directamente a la categoría correspondiente a un patrón de entrada que tienen perfectamente aprendido, independientemente del número y complejidad de las categorías que existan en el sistema.
- *Auto-estabilización*: en respuesta a una secuencia arbitraria de patrones de entrada, el aprendizaje del sistema siempre se estabiliza en un número finito de presentaciones de los patrones de entrada.

2. La Arquitectura ART 1

La arquitectura más básica basada en la Teoría de la Resonancia Adaptativa es la arquitectura ART 1, que está representada en la Fig. 2. El sistema ART 1 está compuesto por dos subsistemas: el subsistema de atención y el subsistema de orientación. El subsistema de atención está compuesto por dos capas: la capa F_1 o capa de las entradas, y la capa F_2 o capa de las categorías. La capa F_1 está formada por N celdas $(v_1, v_2, \ldots, v_N)$ cuyas actividades vienen descritas por el vector $(x_1, x_2, \ldots, x_N)$. La capa de las categorías F_2 está formada por M celdas $(u_1, u_2, \ldots, u_M)$, y su vector de activación es $(y_1, y_2, \ldots, y_M)$. Cada nudo v_i de la capa F_1 está interconectado con cada nudo u_j de la capa F_2 mediante un peso z_{ij}^{bu}. De la misma manera, cada nudo u_j de F_2 está conectado con cada nudo v_i de la capa F_1 mediante un peso z_{ji}^{td}. Los nudos de la capa F_2 están a su vez interconectados entre ellos. Cada nudo de la capa F_2 presenta una realimentación positiva consigo mismo e influye negativamente en los restantes nudos de la capa F_2.

La evolución de un sistema ART 1 viene descrita por dos conjuntos de ecuaciones diferenciales: las ecuaciones STM o de "memoria de corto término", y las ecuaciones LTM o de "memoria de largo término". Las ecuaciones STM describen la evolución de las actividades en los nudos de F_1 y F_2 en función de las interacciones existentes entre ellos. Las ecuaciones LTM describen la evolución de los pesos de interconexión. Las ecuaciones STM evolucionan según una constante de tiempo mucho menor que las ecuaciones LTM. Por ello, es posible suponer que las ecuaciones STM alcanzan su estado estacionario instantáneamente y considerar sólo la dinámica correspondiente a las ecuaciones LTM. Carpenter y Grossberg introdujeron además el modo de "aprendizaje rápido" [10]. En este modo, se supone que las ecuaciones LTM alcanzan también su estado estacionario en cada presentación de un patrón de entrada.

Según el tipo de aproximación que se haga, es posible distinguir tres niveles de implementación del algoritmo ART 1:

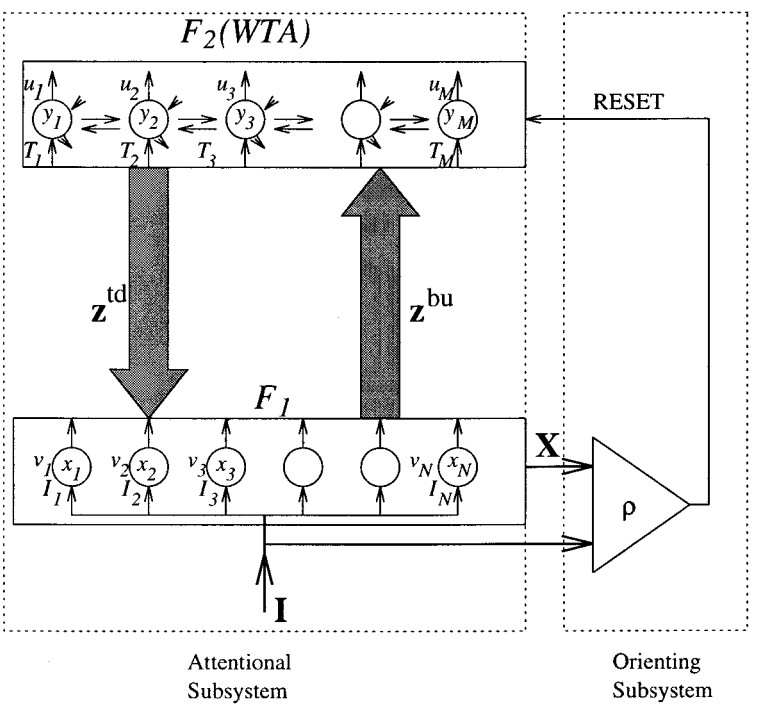


Fig. 2: Diagrama de una Arquitectura ART 1

Tipo-1 *Implementación del Modelo Completo:* Se implementan directamente las ecuaciones diferenciales STM y LTM.

Tipo-2 *Implementación del Estado Estacionario STM:* Se supone que las ecuaciones STM alcanzan el estado estacionario mientras el patrón de entrada **I** permanece constante. Se sustituyen, por tanto, las ecuaciones diferenciales STM por las ecuaciones algebráicas correspondientes al estado estacionario, y sólo se implementan directamente las ecuaciones diferenciales LTM.

Tipo-3 *Implementación de Aprendizaje Rápido:* Tanto las ecuaciones diferenciales STM como las LTM son sustituidas por las ecuaciones algebráicas del estado estacionario. En este caso, hay que implementar artificialmente la secuencia de estados STM y LTM.

La descripción completa del algoritmo ART 1 *Tipo-1* está desarrollada en el Apéndice 1. Aquí describiremos únicamente las ecuaciones de estado estacionario del algoritmo ART 1, esto es, la implementación *Tipo-3*. El diagrama de flujo correspondiente a esta implementación es el que se muestra en la Fig. 3(a).

Todos los pesos z_{ji}^{td} son inicializados al valor '1', mientras que los pesos z_{ij}^{bu} toman inicialmente el valor $\frac{L}{L-1+N}$.

Tal como se observa en la Fig. 2, cada nudo v_i de la capa F_1 recibe una componente del vector de entrada I_i. La entrada T_j que recibe cada nudo u_j de la capa F_2 es el resultado de multiplicar el vector de entrada **I** por todos los pesos z_{ij}^{bu} que conectan los nudos v_i de F_1 con el nudo u_j de F_2. Esto es,

$$T_j = \sum_{i=1}^{N} z_{ij}^{bu} I_i \tag{1}$$

La capa F_2 actúa como un circuito "Winner-Take-All": la realimentación positiva que existe en cada nudo y las inhibiciones laterales entre los distintos nudos hacen que la activación presente en cada nudo tienda a reforzarse y a inhibir la activación en los restantes nudos. En el estado estacionario resulta que solamente el nudo u_j que recibe la entrada T_j máxima permanece activo,

$$y_j = \begin{cases} 1 \text{ if } T_j = max\{T_k\} \\ 0 \text{ en otro caso} \end{cases} \tag{2}$$

Llamemos u_J a la categoría de la capa F_2 que ha recibido la entrada máxima T_J. Una vez que el nudo u_J de F_2 se ha activado, se activará el patrón de pesos z_{Ji}^{td}, modificando la activación **X** de los nudos de la capa F_1. La activación x_i de cada nudo v_i vendrá dada por,

$$x_i = I_i \sum_{j=1}^{M} z_{ji}^{td} y_j = I_i z_{Ji}^{td}, \tag{3}$$

o en notación vectorial

$$\mathbf{X} = \mathbf{I} \cap \mathbf{z}_J^{td}, \tag{4}$$

donde $\mathbf{z}_J^{td} = (z_{J1}^{td}, z_{J2}^{td}, \ldots, z_{JN}^{td})$, $\mathbf{X} = (x_1, x_2, \ldots, x_N)$ e $\mathbf{I} = (I_1, I_2, \ldots, I_N)$.

El subsistema de orientación compara el patrón de activación **X** de los nudos de la capa F_1 con el patrón de entrada original **I**, de acuerdo con un criterio de vigilancia. El criterio de vigilancia está controlado por el parámetro de vigilancia ρ. Tras la comparación pueden ocurrir dos casos:

a) Si $\rho|\mathbf{I}| > |\mathbf{I} \cap \mathbf{z}_J^{td}|$, la categoría u_J no es válida para el nivel de vigilancia impuesto por el parámetro de vigilancia ρ. En este caso u_J será desactivada haciendo $T_J = 0$, de forma que otra categoría $u_{J'}$ se activará por la acción del circuito "Winner-Take-All".[1]

b) Si $\rho|\mathbf{I}| \leq |\mathbf{I} \cap \mathbf{z}_J^{td}|$, la categoría u_J es aceptada y sus pesos se adaptan para incorporar el nuevo conocimiento.

La ley de actualización de los pesos está descrita por las ecuaciones algebráicas siguientes:

$$z_{iJ}^{bu}(new) = \frac{Lx_i}{L-1+|\mathbf{X}|} = \frac{LI_i z_{Ji}^{td}(old)}{L-1+|\mathbf{I} \cap \mathbf{z}_J^{td}(old)|} \tag{5}$$

$$z_{Ji}^{td}(new) = x_i = I_i z_{Ji}^{td}(old) \tag{6}$$

o en notación vectorial,

$$\mathbf{z}_J^{bu}(new) = \frac{L\mathbf{I} \cap \mathbf{z}_J^{td}(old)}{L-1+|\mathbf{I} \cap \mathbf{z}_J^{td}(old)|} \tag{7}$$

$$\mathbf{z}_J^{td}(new) = \mathbf{I} \cap \mathbf{z}_J^{td}(old). \tag{8}$$

La Fig. 3(a) muestra el algoritmo completo de la operación de la arquitectura ART 1 *Tipo-3*.

Obsérvese en las ecuaciones (5) y (6) que los pesos z_{ji}^{td} tomarán siempre valores binarios '1' o '0', mientras que los pesos z_{ij}^{bu} toman valores analógicos. El valor mínimo es '0' (para $x_i = 0$) y el máximo es '1' (para $x_i = 1$ y $|\mathbf{X}| = 1$). A la hora de implementar en circuito el algoritmo mostrado en la Fig. 3(a) surgen dos dificultades. La primera dificultad que aparece es la necesidad de implementar dos conjuntos de pesos: los pesos "bottom-up" z_{ij}^{bu} que pueden tomar cualquier valor comprendido en el intervalo $[0, 1]$, y los pesos "top-down" z_{ji}^{td} que son pesos binarios ya que solamente toman los valores '0' o '1'. Sin embargo, en las ecuaciones (7) y (8) se observa que el conjunto de pesos $\{z_{ij}^{bu}\}$ no es más que una versión normalizada del conjunto de pesos $\{z_{ji}^{td}\}$. Por tanto, se puede implementar un sólo conjunto de pesos binarios $\{z_{ij}\}$ y realizar la normalización durante el cómputo de los términos T_j. En este caso, para calcular los términos T_j habría que modificar la ecuación (1) de forma que se tenga en cuenta la normalización,

---

1. Si **a** es un vector de componentes $(a_1, a_2, \ldots, a_q)$, la notación $|\mathbf{a}|$ se refiere a su norma L_1: $|\mathbf{a}| = \sum_{i=1}^{q} |a_i|$

$$T_j = \frac{LT_{Aj}}{L-1+T_{Bj}} = \frac{L\sum_{i=1}^{N} z_{ij}I_i}{L-1+\sum_{i=1}^{N} z_{ij}} = \frac{L|\mathbf{z}_j \cap \mathbf{I}|}{L-1+|\mathbf{z}_j|}. \tag{9}$$

El resultado de introducir esta pequeña modificación es el algoritmo representado en la Fig. 3(b). La operación a nivel de sistema del algoritmo de la Fig. 3(a) es idéntica a la de la Fig. 3(b). Sin embargo, el segundo no requiere realizar físiscamente pesos analógicos.

Desafortunadamente, ahora surge la dificultad de implementar la operación de división necesaria en el cálculo de los términos T_j. Esta es una operación muy costosa de realizar por circuitos tanto para una implementación analógica como digital. Esto nos ha llevado a sustituir la operación de división en el cálculo de los términos T_j por una operación de sustracción. La operación de sustracción se realiza fácilmente con circuitos sin más que aplicar la ley de Kirchoff de las intensidades en un nudo del circuito. De esta forma, el cálculo de los elementos T_j se reduce a calcular,

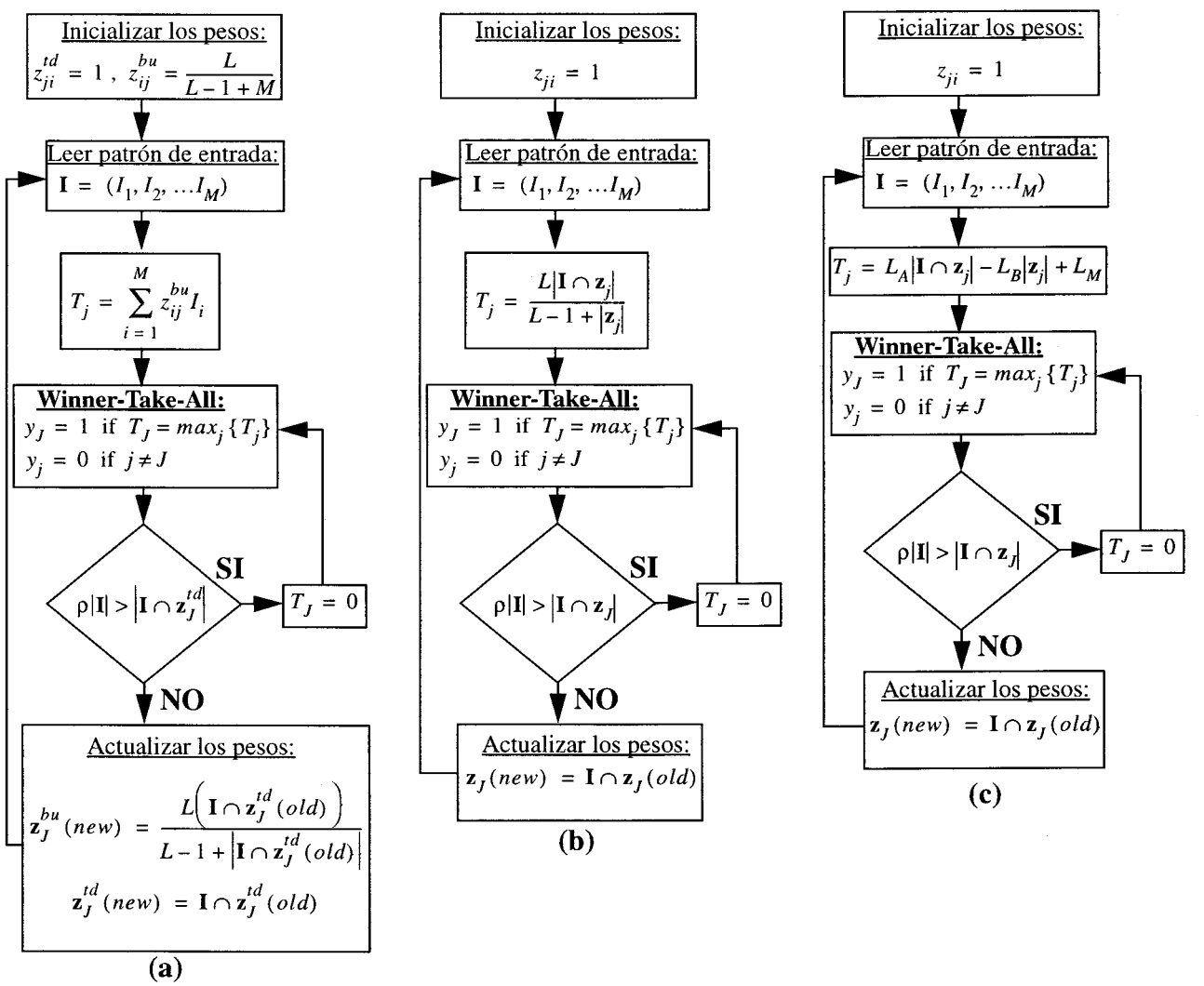


Fig. 3: Implementación *Tipo-3* del Algoritmo de la Arquitectura ART 1: (a) ART 1 original (b) ART 1 con una sola matriz de pesos binarios (c) y ART 1 modificado para la implementación VLSI

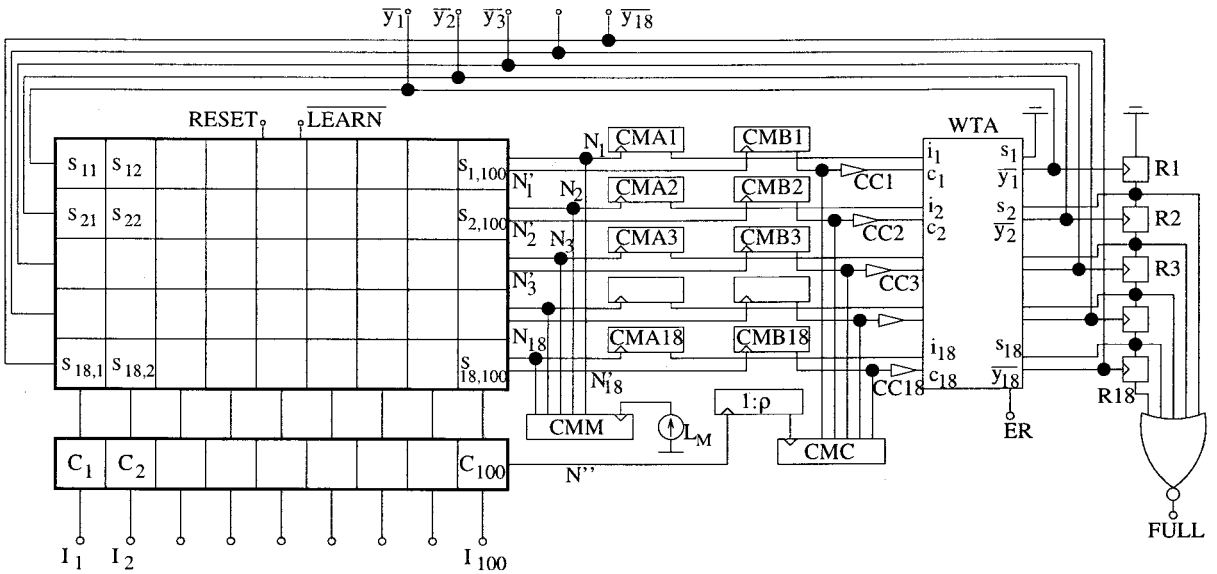


Fig. 4: Diagrama de Bloques del Circuito que Implementa el Algoritmo ART 1 Modificado

$$T_j = L_A T_{Aj} - L_B T_{Bj} + L_M \tag{10}$$

donde L_A y L_B son parámetros siempre positivos que realizan el papel de los parámetros L y $L-1$ del algoritmo original. L_M es un parámetro también positivo que se introduce para asegurar que todos los elementos T_j son siempre positivos.

El algoritmo que resulta tras esta nueva modificación está representado en la Fig. 3(c). En el Apéndice 1, se compara la operación a nivel de sistema del algoritmo original y del algoritmo resultante tras esta modificación. Así mismo se demuestra que el algoritmo modificado conserva todas las propiedades del algoritmo original. Sin embargo, el algoritmo modificado presenta unas propiedades mucho más interesantes desde el punto de vista de la implementación con circuitos.

3. Implementación de Circuito del Algoritmo ART 1

La Fig. 4 muestra un diagrama de bloques de un circuito que implementa el algoritmo de la Fig. 3(c). El circuito consiste en un array de 18×100 sinapsis S_{11}, ...,$S_{18,100}$, un vector de 100 celdas de entrada C_1, ...,C_{100}, dos arrays de 1×18 espejos de corriente de ganancia unidad $CMA1$, ..., $CMA18$, $CMB1$, ...$CMB18$, un array de 1×18 comparadores de corriente $CC1$, ..., $CC18$, un circuito WTA de 18 entradas, dos espejos de corriente de 18 salidas de ganancia unidad CMM y CMC, y un espejo de corriente de ganancia variable ρ.

Los diagramas de cada sinapsis S_{ij} y de las celdas de entrada C_i están detallados en la Fig. 5(a) y en la Fig. 5(b), respectivamente. Cada sinapsis S_{ij} contiene una celda de memoria que almacena el valor del peso z_{ij}, dos fuentes de corriente L_A, una fuente de corriente L_B, más circuitería para controlar las corrientes de salida y para el aprendizaje de los pesos. Las señales "RESET" y "LEARN" son señales de control compartidas por todas las sinapsis del circuito. Cada sinapsis genera dos corrientes de salida:

- Una corriente de valor $L_A I_i z_{ij} - L_B z_{ij}$ que entra en el espejo CMA_j.
- Una corriente $L_A I_i z_{ij}$ que entra en el espejo CMB_j.

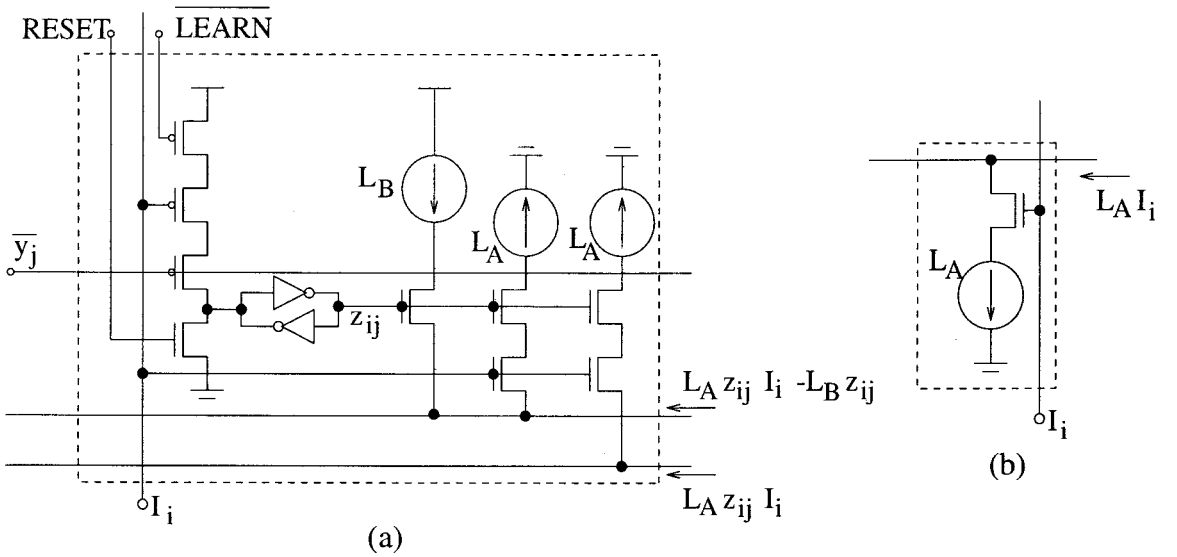


Fig. 5: (a) Detalles del Circuito de la Sinapsis S_{ij}, (b) Detalles del Circuito de la Fuente de Corriente Controlada C_i

El espejo CMA_j recibe todos los elementos de corriente $L_A I_i z_{ij} - L_B z_{ij}$ de las sinapsis de la fila j además de una corriente L_M que es replicada por el espejo CMM. De esta forma la corriente de entrada que distribuye CMA_j al WTA vale

$$T_j = L_A \sum_i I_i z_{ij} - L_B \sum_i z_{ij} + L_M = L_A |\mathbf{I} \cap \mathbf{z}_j| - L_B |\mathbf{z}_j| + L_M \tag{11}$$

El espejo CMB_j recibe todos los elementos de corriente $L_A I_i z_{ij}$ de las sinapsis de la fila j,

$$L_A \sum_i I_i z_{ij} = L_A |\mathbf{I} \cap \mathbf{z}_j| . \tag{12}$$

Cada celda de entrada C_i genera una corriente $L_A I_i$. Todas estas corrientes generadas por las celdas C_i se suman en la entrada del espejo de ganancia variable ρ. De esta forma, la corriente distribuida por el espejo CMC a la entrada de los comparadores de corriente es

$$\rho L_A \sum_i I_i = \rho L_A |\mathbf{I}| . \tag{13}$$

Cada comparador CC_j recibe una corriente de entrada total $L_A |\mathbf{I} \cap \mathbf{z}_j| - \rho L_A |\mathbf{I}|$ y la compara frente a '0'. Si esta corriente es positiva, las señales de control c_j del WTA están altas. Cuando la señal c_j está alta la corriente T_j entra a formar parte de la competición del WTA. Por el contrario, si la corriente de entrada del comparador CC_j es negativa, la señal c_j está baja lo que impide a la corriente T_j participar en la competición del WTA.

El circuito de la Fig. 4 sigue la siguiente secuencia de operación:

1.- La señal de "RESET" es activada. Todos los pesos z_{ij} son inicializados al valor '1'.

2.- Se presenta un patrón de entrada **I**.

3.- Las 18 filas de sinapsis generan las corrientes de entrada a las 18 celdas del WTA:
$T_j = L_A |\mathbf{I} \cap \mathbf{z}_j| - L_B |\mathbf{z}_j| + L_M$ y las 18 corrientes de entrada a los comparadores $L_A |\mathbf{I} \cap \mathbf{z}_j|$.

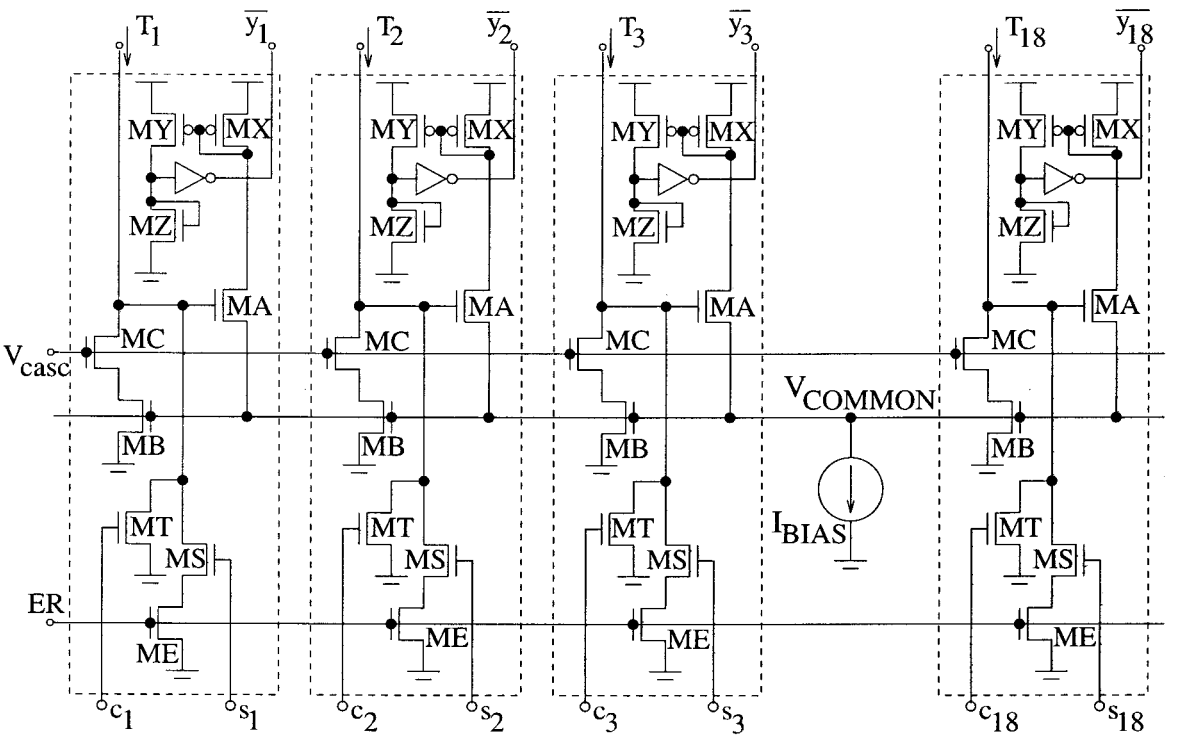


Fig. 6: Esquemático del Circuito Winner-Take-All (WTA)

4.- La fila de celdas de entrada C_i genera la corriente $L_A|\mathbf{I}|$ que es multiplicada por ρ y distribuida por el espejo *CMC* a los 18 comparadores.

5.- Cada comparador compara la corriente $L_A|\mathbf{I} \cap \mathbf{z}_j| - \rho L_A|\mathbf{I}|$ frente a '0':

- Si dicha corriente es negativa, la corriente T_j es desviada y no entra en el WTA para la competición.
- Si la corriente de entrada al comparador es positiva, la corriente T_j entra en la competición del WTA.

6.- El WTA selecciona un ganador para el que se hace $\overline{y_J} = 0$.

7.- Una vez que se ha elegido un único ganador, se activa la señal "LEARN" ($\overline{\text{LEARN}}$=0). Los pesos z_{iJ} correspondientes a la categoría seleccionada se modifican de forma que se hace '0' para aquellas sinapsis en las que $I_i = 0$.

En el Apéndice 2 se explican los detalles de un prototipo fabricado en la tecnología de ES2-1.5μm y se incluyen resultados experimentales.

A. Realización del Circuito "Winner-Take-All"

La operación de selección de categorías se realiza con el circuito mostrado en la Fig. 6. El circuito está basado en el WTA reportado por Lazzaro [16]. En el circuito hay un array de transistores MB por los que fluyen las corrientes de entrada T_j, y un array de transistores MA que se reparten la corriente de polarización I_{BIAS}. Todos los transistores MB comparten sus puertas en el nudo V_{COMMOM} y tienen las fuentes a tierra. La precisión de la operación de este circuito depende del apareamiento entre todos los transistores MA y todos los transistores MB. Por tanto, esta precisión se verá seriamente degradada cuando intentemos distribuir el

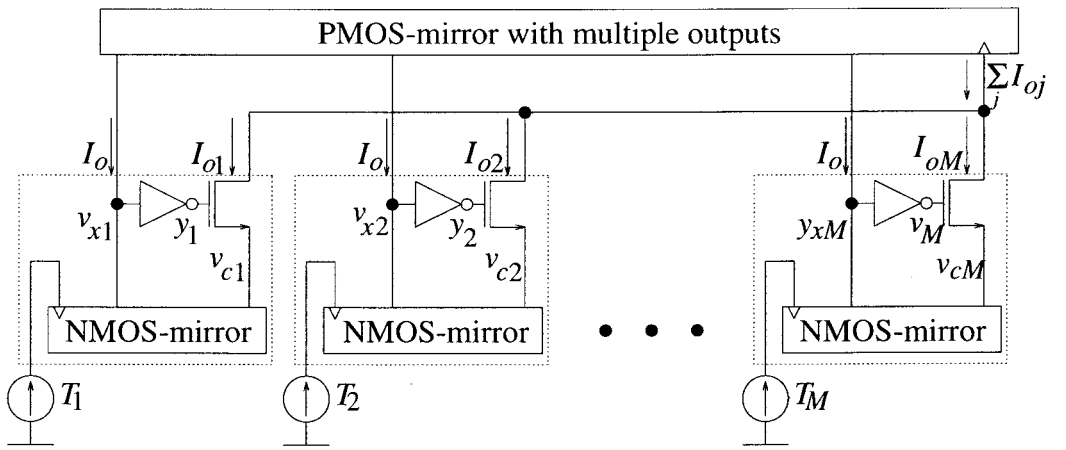


Fig. 7: Diagrama del Circuito WTA en Modo de Corriente

circuito entre distintos chips. Una solución para este problema es usar un WTA cuya operación se base únicamente en el reflejo y comparación de corrientes. En el Apéndice 3 se describe una topología de un circuito WTA que opera totalmente en modo de corriente. La operación de este circuito se puede distribuir entre distintos chips sin pérdida significativa de precisión si se reproducen adecuadamente las corrientes de un chip a otro.

La Fig. 7 muestra el diagrama del circuito WTA en modo de corriente. El circuito consta de M celdas interconectadas entre sí mediante un espejo PMOS de M salidas. Cada celda del circuito consta de un espejo NMOS de dos salidas, un transistor NMOS y un comparador de corriente. Cada celda recibe dos corrientes de entrada: T_j e I_o. T_j es la corriente de entrada aplicada externamente que es replicada dos veces mediante el espejo NMOS. I_o es una corriente suministrada por una de las salidas del espejo PMOS. Cada celda da una corriente de salida I_{oj}. Todas las corrientes de salida I_{oj} se suman en el nudo de entrada del espejo PMOS y dicha suma $I_o = \sum I_{oj}$ es distribuida por el espejo PMOS a la entrada de todas las celdas. El comparador de corriente de cada celda compara la diferencia de las entradas $I_o - T_j$ frente a '0'. Si esa diferencia es positiva la salida del comparador estará baja y la corriente de salida es $I_{oj} = 0$. Por el contrario, si la diferencia $I_o - T_j$ es negativa la salida del comparador estará alta y el transistor NMOS conducirá la corriente de salida $I_{oj} = T_j$.

La salida de la celda j del WTA la podemos, por tanto, expresar como

$$I_{oj} = \begin{cases} T_j \text{ si } T_j \geq I_o \\ 0 \text{ si } T_j < I_o \end{cases}, \tag{14}$$

o de forma más compacta,

$$I_{oj} = T_j H(T_j - I_o) \tag{15}$$

donde la función $H(x)$ es la función escalón que se define como

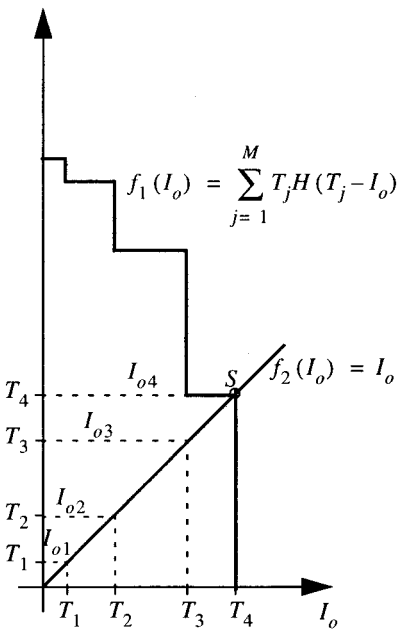


Fig. 8: Representación Gráfica de la Solución de la Ecuación (18)

$$H(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases} . \tag{16}$$

Por otro lado, podemos expresar la corriente I_o como

$$I_o = \sum_{j=1}^{M} I_{oj} . \tag{17}$$

De las ecuaciones (15) y (17) se deduce que

$$I_o = \sum_{j=1}^{M} T_j H(T_j - I_o) . \tag{18}$$

La Fig. 8 muestra la representación gráfica de las funciones $f_1(I_o) = \sum T_j H(T_j - I_o)$ y $f_2(I_o) = I_o$. El punto de intersección de ambas funciones será la solución de la ecuación del circuito (18). Se observa que existe un único punto de equilibrio y que el valor de I_o en el punto de equilibrio es tal que

$$I_o|_{eq} = max\{T_j\} . \tag{19}$$

En el punto de equilibrio la única celda que da una corriente de salida no nula $I_{oj} = T_j$ es la celda ganadora cuya corriente de entrada T_j es máxima. Por lo tanto, el circuito realiza la operación de un WTA.

La precisión de la operación del circuito está determinada básicamente por la precisión en la reflexión de las corrientes, ya que el offset del comparador de corriente es despreciable. Por tanto, si las corrientes se reflejan con precisión de un chip a otro, la operación se puede distribuir entre varios chips sin sufrir una pérdida de precisión significativa.

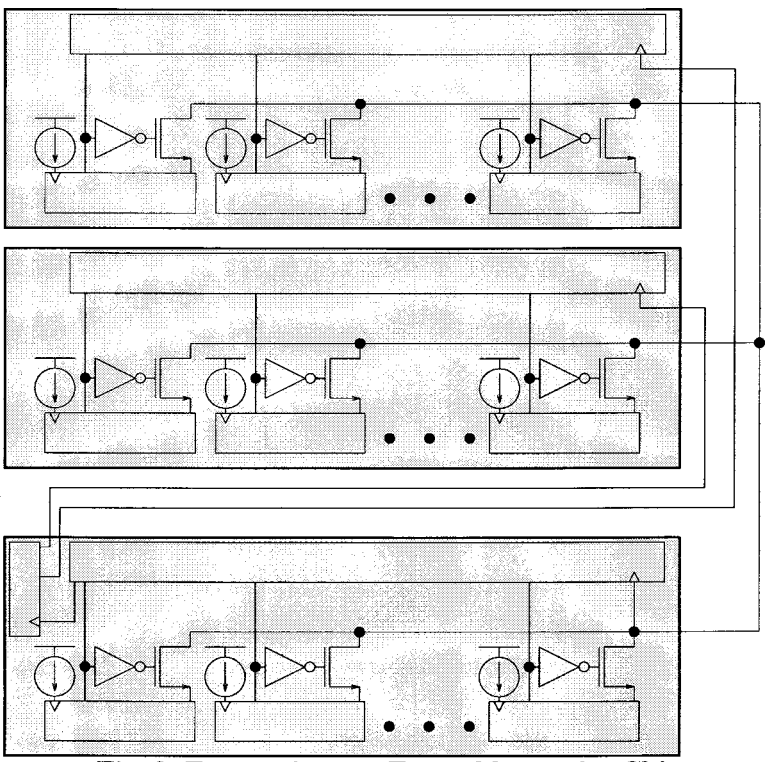


Fig. 9: Estrategia para Ensamblar varios Chips

La Fig. 9 muestra un diagrama de la estrategia para ensamblar varios WTA. El ensamblaje entre los chips es posible sin más que añadir una reflexión adicional en un espejo de corriente. Se observa que el nudo de suma de las corrientes I_{oj} es compartido por todos los chips. El espejo NMOS adicional distribuye la corriente global I_o entre las entradas de los espejos PMOS de los distintos chips.

El WTA en modo de corriente ha sido fabricado y testado para dos tecnologías diferentes: la tecnología CMOS de poly simple y doble metal de ES2-1.0μm y la tecnología CMOS con doble metal y doble poly de MIETEC-2.4μm. En el Apéndice 3 se discuten los detalles de la implementación del circuito y se incluyen los resultados experimentales de los chips y de un sistema formado por la interconexión de dos chips; cuando los chips son de la misma tecnología y cuando cada uno de ellos es de una tecnología diferente. Se demuestra que la operación de los circuitos es correcta y que la precisión no se degrada significativamente al distribuir el sistema entre diferentes chips, ni siquiera si éstos son de tecnologías diferentes.

B. Realización de los Elementos de Corriente

Para la realización de las fuentes de corriente L_A y L_B de las sinapsis S_{ij} del circuito de la Fig. 4, y de las celdas de entrada C_i es necesario replicar la corriente L_A un total de $2 \times 1800 + 100 = 3700$ veces, y la corriente L_B debe ser replicada 1800 veces. Además, estas fuentes de corriente están distribuidas en un área de aproximadamente $1 cm^2$. Para mantener una precisión del orden del 1% hemos replicado las corrientes utilizando una estructura de árbol de espejos de corriente tal como se muestra en la Fig. 10 para las corrientes L_B. Cada etapa es un espejo de corriente de salida múltiple. Los transistores de cada etapa del espejo tienen una geometría de $W = L = 10\mu m$ y están realizados utilizando técnicas de layout centroide común para mejorar el apareamiento entre los transistores.

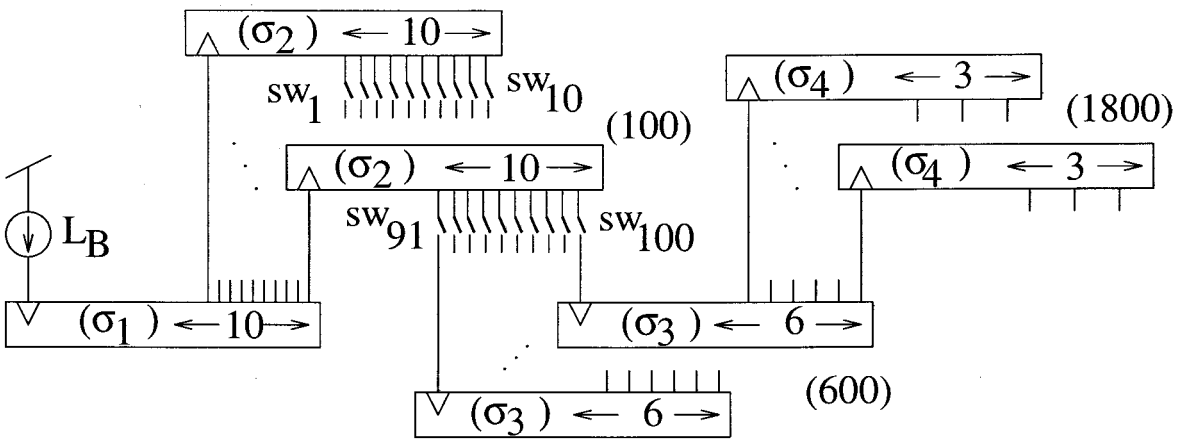


Fig. 10: Cascada de Espejos de Corriente para Reducir el Error de Desapareamiento

La desviación estándar medida en las corrientes de salida L_A de varias sinapsis es menor que 1% para corrientes de operación mayores que $5\mu A$.

Para mantener un nivel de precisión adecuado en el diseño de los espejos de corriente sin incurrir en un derroche de área por miedo al "mismatching" es necesario caracterizar el mismatching del proceso tecnológico que se esté empleando. Con objeto de simplificar el diseño de las fuentes de corriente L_A y L_B de las sinapsis hemos caracterizado estadísticamente las desviaciones en los parámetros de los transistores del proceso CMOS de ES2-1.0μm. El método de caracterización empleado es válido para la caracterización estadística del mismatching de cualquier proceso tecnológico CMOS.

C. Caracterización del "Mismatching" de una Tecnología CMOS

Según el modelo de Pelgrom [17], la desviación estándar de la diferencia del parámetro eléctrico P entre dos transistores viene dada por una ecuación de la forma

$$\sigma^2(\Delta P) = \frac{A_P^2}{WL} + S_P^2 D^2 \tag{20}$$

donde WL es el área de los transistores, D la distancia entre ambos transistores, y A_P y S_P son parámetros característicos del proceso tecnológico.

En la ecuación (20) se distinguen dos componentes que influyen en la desviación de un parámetro: una componente aleatoria o de ruido dependiente del área de los transistores caracterizada por el parámetro A_P, y otra componente de gradiente dependiente de la distancia entre los transistores caracterizada por el parámetro S_P.

El parámetro A_P es muy estable de chip a chip, de "run" a "run", de "wafer" a "wafer" e incluso entre distintas "foundries" que utilizan procesos tecnológicos parecidos. Puede, por tanto, ser caracterizada con muy pocas muestras. Sin embargo, el parámetro S_P necesita muchas muestras para su caracterización. Además, la componente de error dependiente de la distancia puede ser eliminada con técnicas de layout [18]

por lo que su caracterización no tiene tanto interés para el diseñador. Nosotros hemos caracterizado solamente los parámetros A_P.

Para la caracterización de un proceso CMOS empleamos un chip de propósito específico realizado en dicho proceso. El circuito está formado por una matriz de celdas. Cada celda contiene pares de transistores NMOS y PMOS de diferentes tamaños. El chip contiene además circuitería de decodificación para seleccionar en cada momento un solo par de transistores. La Fig. 11 muestra un esquema simplificado del chip y del equipo de medida. En el chip, todos los transistores NMOS comparten su drenador en el pin DN. Todos los transistores PMOS comparten sus drenadores en el pin DP. Todos los transistores NMOS y PMOS comparten sus fuentes en el pin S. En cada momento, solamente los transistores NMOS y PMOS del par seleccionado tienen sus puertas conectadas al pin G. Los restantes transistores del chip tienen sus puertas cortocircuitadas con sus terminales de fuente. Si el pin DP se deja desconectado y se accede a los terminales DN, G, y S se puede caracterizar el transistor NMOS del par seleccionado. De la misma manera, dejando desconectado el pin DN y accediendo a los terminales DP, G y S es posible caracterizar el transistor PMOS del par seleccionado.

El par de transistores activo se selecciona a través de un bus digital controlado por un ordenador exterior, el cual controla a su vez un equipo trazador de curvas en DC (HP4145) que mide las curvas para la caracterización de los transistores del par seleccionado.

Para la caracterización de cada transistor se miden dos curvas correspondientes a zona óhmica:

- Curva 1: $V_{DS} = 0.1V$, $V_{SB} = 0V$

  $V_{GS} = 1.5V\text{-}5.0V$

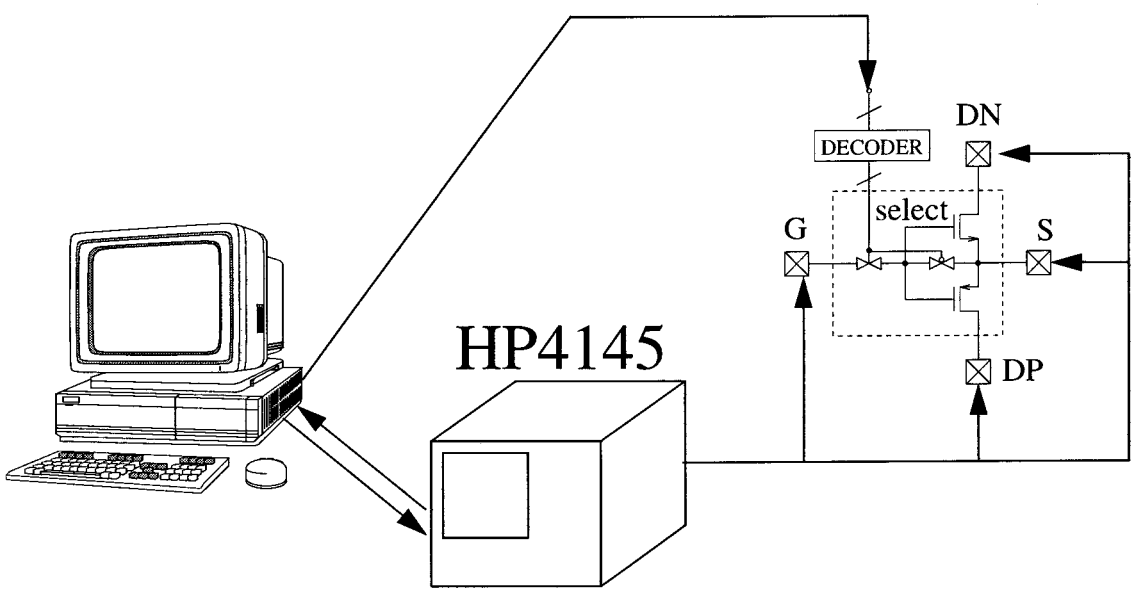


Fig. 11: Montaje Experimental para la Caracterización Automática del "Mismatching" entre Transistores MOS

$$I_{DS} = \beta \frac{\left(V_{GS} - V_{T0} - \frac{1}{2}0.1V\right)0.1V}{1 + \theta\,(V_{GS} - V_{T0})}$$

- Curva 2: $V_{DS} = 0.1V$

$$V_{GS} = 3.0V$$

$$V_{SB} = 0.0V\text{-}2.0V$$

$$I_{DS} = \beta \frac{\left(3.0V - V_T(V_{SB}) - \frac{1}{2}0.1V\right)0.1V}{1 + \theta\,(3.0V - V_T(V_{SB}))}$$

Estas curvas se ajustan utilizando técnicas de ajuste no lineal a las curvas de nivel 1 del modelo de HSPICE de un transistor en zona óhmica

$$I_{DS} = \beta \frac{\left(V_{GS} - V_T(V_{SB}) - \frac{1}{2}V_{DS}\right)V_{DS}}{1 + \theta\,(V_{GS} - V_T(V_{SB}))} \qquad V_{DS} \le V_{GS} - V_T \tag{21}$$

$$V_T(V_{SB}) = V_{T0} + (\eta - 1)\,V_{SB} + \gamma\left(\sqrt{\phi + V_{SB}} - \sqrt{\phi}\right) \tag{22}$$

En el proceso de ajuste los parámetros η y θ se consideran constantes por lo que se ajustan sólo para el primer transistor. Los parámetros β, V_{TO} y γ se extraen para todos los transistores del chip.

Para cada parámetro P, para cada geometría de los transistores S, para cada tipo de transistor K (NMOS o PMOS) y para cada chip, se calcula la desviación estándar de las diferencias entre los parámetros de los transistores situados consecutivamente a lo largo de la dirección x y a lo largo de la dirección y

$$\Delta_x P_{SK}(n, m) = P_{SK}((n+1)\Delta x, m\Delta y) - P_{SK}(n\Delta x, m\Delta y) \qquad \begin{cases} n = 0, \ldots n^{max} - 1 \\ m = 0, \ldots m^{max} \end{cases} \tag{23}$$

$$\Delta_y P_{SK}(n, m) = P_{SK}(n\Delta x, (m+1)\Delta y) - P_{SK}(n\Delta x, m\Delta y) \qquad \begin{cases} n = 0, \ldots n^{max} \\ m = 0, \ldots m^{max} - 1 \end{cases} \tag{24}$$

A continuación se calcula la desviación estándar relativa de las diferencias del parámetro,

$$\sigma_{P,SK} = \sqrt{\frac{\sigma^2(\Delta_x P_{SK}) + \sigma^2(\Delta_y P_{SK})}{2}}. \tag{25}$$

Ajustando, en cada chip, los valores de las desviacion obtenidas para las distintas geometrías a una curva del tipo

$$\sigma_{P,SK} = \frac{A_P}{\sqrt{W_{eff} L_{eff}}} \qquad \begin{cases} W_{eff} = W - WD \\ L_{eff} = L - LD \end{cases} \tag{26}$$

se obtiene el valor del parámetro A_P en cada chip. Promediando para todos los chips se obtiene el valor de A_P de la tecnología. En el Apéndice 4 se muestran los resultados de caracterización para la tecnología de ES2-1.0μm y para la tecnología del CNM-2.5μm.

Usando esta técnica de caracterización de los transistores hemos obtenido los parámetros $\beta(x, y)$, $V_{TO}(x, y)$ y $\gamma(x, y)$ de una matriz de 6×6 celdas fabricada en la tecnología de ES2-1.0μm, que ocupa un área total de $5.5cm^2$. Para cada posición del transistor, podemos calcular la desviación de cada parámetro con respecto al valor medio

$$\frac{\Delta\beta(x, y)}{\bar{\beta}} = \frac{\beta(x, y) - \bar{\beta}}{\bar{\beta}} \tag{27}$$

$$\Delta V_{TO}(x, y) = V_{TO}(x, y) - \overline{V_{TO}} \tag{28}$$

$$\Delta\gamma(x, y) = \gamma(x, y) - \bar{\gamma}. \tag{29}$$

Usando estas desviaciones podemos construir un fichero de entrada a HSPICE donde los transistores cuyos parámetros hemos extraído son los transistores de salida de un espejo de corriente de salida múltiple. De esta forma, obtenemos la superficie de las corrientes de salida simuladas $I_o^s(x, y)$. Para cada una de estas superfices calculamos el plano $I_o^p(x, y) = Ax + By + C$ que mejor se ajusta a la superficie simulada $I_o^s(x, y)$.

La componente aleatoria de las corrientes de salida simuladas $I_o^s(x, y)$ la calculamos como la desviación estándar de la diferencia $\Delta I_o(x, y) = I_o^s(x, y) - I_o^p(x, y)$. Esto es,

$$\sigma(\Delta I_o) = \sqrt{\frac{\sum_{n=1}^{n^{max}} \sum_{m=1}^{m^{max}} \left(I_o^s(n\Delta x, m\Delta y) - I_o^p(n\Delta x, m\Delta y) - \overline{\Delta I_o} \right)^2}{n^{max} \times m^{max}}} \tag{30}$$

donde n^{max} y m^{max} son el número de veces que cada transistor está repetido en las direcciones x e y, y Δx y Δy son las distancias entre dos transistores de celdas consecutivas en las direcciones x e y.

La máxima desviación sistemática en las corrientes de salida la calculamos como

$$\Delta I_o^p = I_{maxplane} - I_{minplane} \tag{31}$$

donde $I_{maxplane}$ e $I_{minplane}$ son el máximo y el mínimo del plano de interpolación.

Teniendo en cuenta que el 98% de los valores aleatorios se mantienen en un intervalo de $\pm 3\sigma(\Delta I_o)$, la relación entre la componente de error aleatorio y la componente de error sistemático vendrá dada por

$$r^{ran/sys} = \frac{6 \times \sigma(\Delta I_o)}{\Delta I_o^p} \tag{32}$$

Realizando las simulaciones para los transistores de geometría $W = L = 10\mu m$, medidos en el array de 6×6 celdas, para niveles de corriente de $10\mu A$ obtuvimos que la componente de error sistemático era del mismo orden (y generalmente menor) que la componenete aleatoria. El Apéndice 5 contiene un resumen de los resultados obtenidos en esas simulaciones.

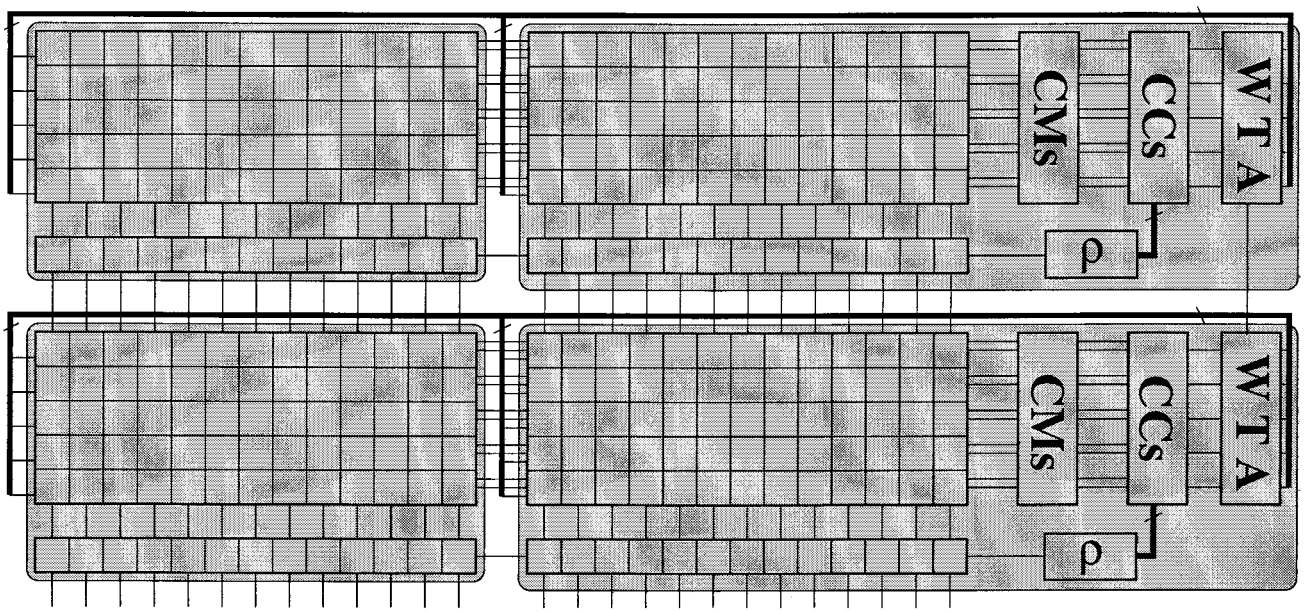


Fig. 12: Conexión entre Chips para la Expansión Modular del Sistema

D. Nuevo Prototipo ART 1

Basándonos en estos resultados, diseñamos otro prototipo del ART 1 que contiene una matriz de 50×10 sinapsis. La matriz de sinapsis ocupa $2.1 mm^2$. La diferencia fundamental con respecto al primer prototipo es que las fuentes de corriente L_A y L_B de las sinapsis no están realizadas mediante un árbol de espejos de corriente. Las fuentes de corriente L_A y L_B de las diferentes sinapsis se realizan directamente mediante un espejo de salida múltiple. Con esta modificación se consigue reducir 15 veces el área del primer prototipo. Los resultados experimentales de las corrientes de salida medidas en todas las sinapsis están contenidos en el Apéndice 5. Para un nivel de corriente de $10\mu A$, la desviación total de las corrientes de salida (debida a las componentes aleatoria y sistemática) es siempre inferior al 1%. El Apéndice 5 contiene además resultados del comportamiento a nivel de sistema del nuevo prototipo.

4. Realizaciones Multichip

En el Apéndice 5 se muestran resultados experimentales de dos tipos de realizaciones multichip utilizando el prototipo ART 1 de 50×10 sinapsis:

- Una realización en la que se han interconectado dos chips horizontalmente para aumentar el número de entradas.
- Una implementación de un sistema ARTMAP de aprendizaje supervisado utilizando dos módulos ART 1 interconectados mediante un módulo inter-ART.

A. Expansión Modular del Sistema

La expansión modular del sistema es posible tanto horizontal como verticalmente. La conexión horizontal de N chips permite aumentar el número de pixels de entrada a $N \times 50$. Conectando verticalmente M chips

aumentamos el número de categorías del sistema a $M \times 10$. La Fig. 12 muestra el diagrama para la interconexión de varios chips.

Para interconectar verticalmente varios chips es necesario interconectar los circuitos WTA de los distintos chips para la elección de un único ganador.

La interconexión horizontal requiere compartir los nudos N_j y N_j' donde se realiza la suma de las corrientes de las sinapsis de la fila j. También el nudo N" debe ser compartido entre los distintos chips para realizar la suma de las corrientes de salida de las celdas C_i. En la interconexión horizontal de varios chips solamente uno de los WTA permanece activo. En los restantes chips, se desconectan los nudos de suma de las corrientes N_j N_j' y N" de las entradas a los espejos $CMA1, \ldots, CMA10, CMB1, \ldots CMB10$ y CMC. Las salidas y_j del WTA activo deben ser compartidas por todas las sinapsis de la fila j de los distintos chips para implementar la regla de aprendizaje.

En el Apéndice 5 se detallan resultados experimentales de la interconexión horizontal de dos chips.

B. Realización de un Sistema ARTMAP

Una arquitectura ARTMAP es un sistema que puede ser entrenado para aprender de forma supervisada la correspondencia entre patrones de entrada binarios.

La Fig. 13 muestra un diagrama de bloques de una arquitectua ARTMAP. El sistema está compuesto por dos módulos ART 1 interconectados por medio de un módulo inter-ART. La letra a denota a los elementos del sistema $ART\ 1^a$ y el subíndice b denota a los elementos del sistema $ART\ 1^b$.

Los M_b nudos del módulo inter-ART están interconectados uno a uno con los M_b nudos de la capa F_2^b mediante un vector de pesos bidireccionales que tienen un valor constante igual a '1'. Los M_a nudos de la

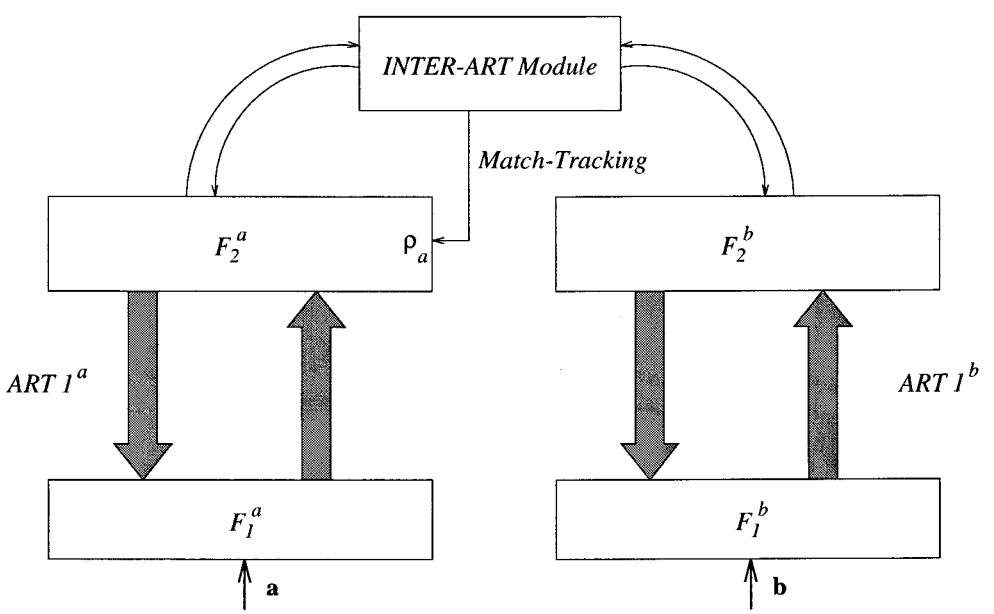


Fig. 13: Arquitectura ARTMAP

capa F_2^a están totalmente interconectados con los M_b nudos del módulo inter-ART mediante una matriz de pesos binarios que son inicializados a '1'.

El sistema puede operar en dos modos:

- Modo de entrenamiento: Durante el cual se le presentan al sistema secuencias de pares de patrones de entrada **a** y **b** cuya correspondencia ha de aprender.

  El módulo $ART\ 1^a$ inicia un proceso de búsqueda durante el cual clasifica al patrón **a** en una categoría u_J^a que satisface el criterio de vigilancia impuesto por ρ_a. De la misma manera, el módulo $ART\ 1^b$ clasifica al patrón **b** en una categoría u_K^b que satisface el criterio de vigilancia impuesto por ρ_b.

  Si el peso w_{JK} de interconexión entre la categoría u_J^a y el nudo u_K^{ab} del módulo inter-ART tiene un valor '1' el sistema aprende la correspondencia entre las categorías activadas en F_2^a y F_2^b. El peso w_{JK} correspondiente a las dos categorías activadas u_J^a y u_K^b se mantiene a '1'. Todos los pesos w_{Jk} que establecen la correspondencia entre la categoría activada u_J^a en F_2^a y las categorías no activadas en F_2^b almacenarán un valor '0'.

  Si, por el contrario, el peso w_{JK} que corresponde a la categoría u_J^a activada en F_2^a y la categoría u_K^b activada en F_2^b vale '0', significa que se ha producido un error de predicción. La categoría u_J^a había aprendido previamente a predecir una categoría de F_2^b distinta de u_K^b. En este caso, el parámetro de vigilancia del sistema $ART\ 1^a$ es aumentado hasta el mínimo valor necesario para resetear la categoría u_J^a. El proceso continúa hasta que se activa una categoría de F_2^a que predice la categoría u_K^b activada en F_2^b o bien que no ha aprendido previamente a predecir ninguna categoría de F_2^b. En cualquiera de los dos casos se encuentra una categoría u_J^a tal que $w_{JK} = 1$.

  La secuencia de operación del algoritmo ARTMAP durante el entrenamiento está representada en la Fig. 14. En ella la operación del los módulos ART 1 se supone que está descrita por el algoritmo ART 1 modificado.

- Modo de predicción. Durante el modo de predicción se le presenta al sistema un único patrón de entrada **a**.

  El módulo $ART\ 1^a$ inicia una búsqueda tras la que clasifica el patrón de entrada en la categoría u_J^a tal que la entrada T_J^a es máxima y satisface el criterio de vigilancia impuesto por ρ_a.

  La categoría u_K^b tal que el peso correspondiente w_{JK} sea igual a '1', será la categoría de F_2^b predicha por el sistema.

  La Fig. 15 muestra el diagrama de flujo del modo de predicción del algoritmo ARTMAP.

Para la realización en modo de circuito del algoritmo ARTMAP se han interconectado dos chips ART 1 mediante un módulo inter-ART, tal como se muestra en la Fig. 16. La Fig. 17 muestra el esquemático del módulo inter-ART. Consiste en una matriz de $M_a \times M_b$ celdas c_{jk}. Cada celda c_{jk} recibe dos señales de entrada y_j^a y y_k^b, y almacena el valor del peso correspondiente w_{jk}. Las señales de "RESET" y "LEARN" son comunes a todas las celdas del circuito. El chip incluye también circuitería adicional para leer el peso w_{JK} correspondiente a las categorías u_J^a y u_K^b cuyas señales de salida y_J^a e y_K^b están activadas.

La activación de la señal de "RESET" o de inicialización del sistema hace que todos los pesos w_{jk} almacenen el valor '1'.

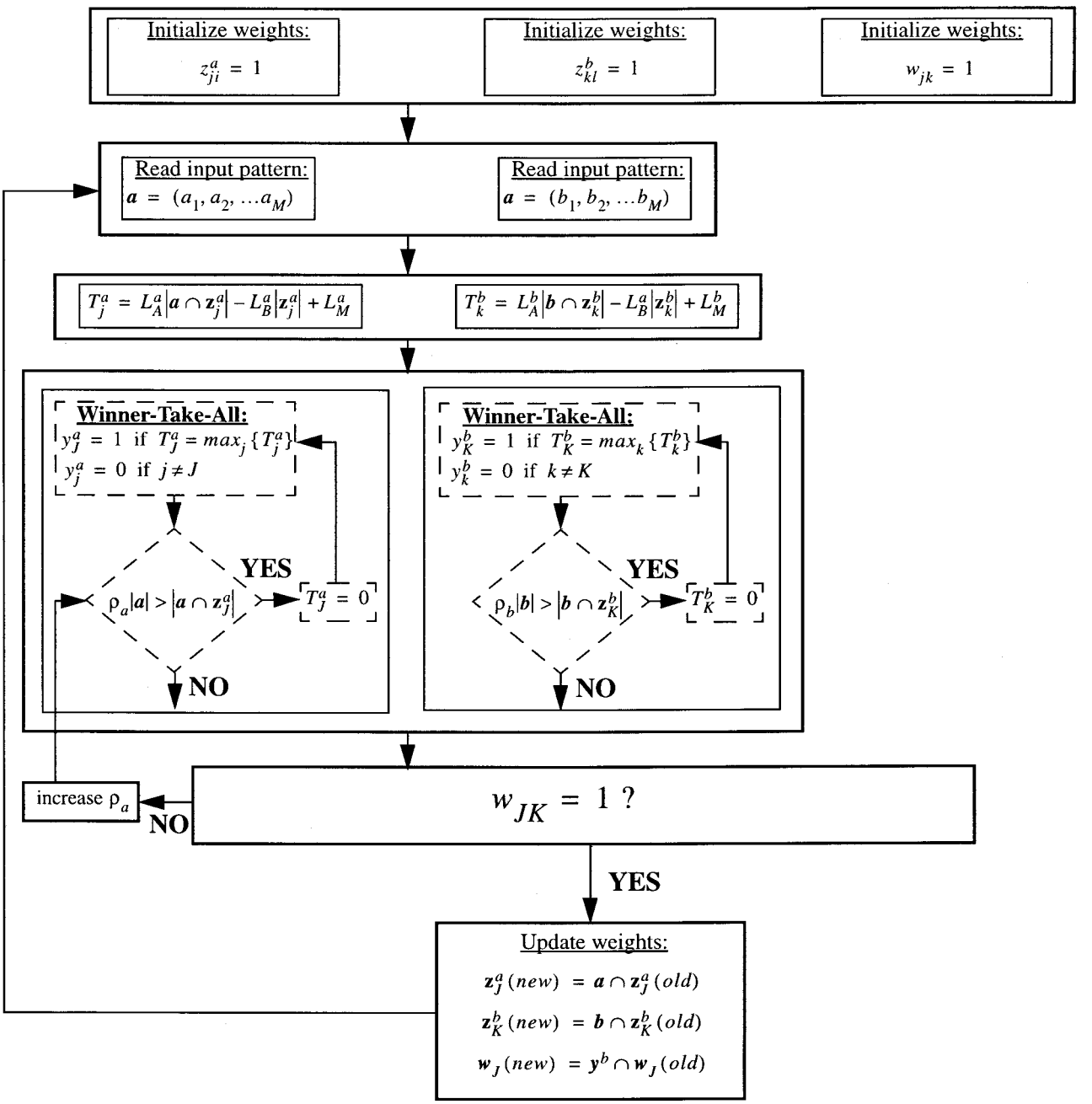


Fig. 14: Diagrama de Flujo de la Operación ARTMAP durante la Fase de Entrenamiento

La activación de la señal de "LEARN" hace que todos los pesos w_{Jk} correspondientes a la categoría activada u^a_J se hagan '0' excepto aquel para el cual la salida y^b_K de la categoría u^b_K vale '1'.

En el Apéndice 5 están contenidos los resultados experimentales del comportamiento a nivel de sistema del circuito ARTMAP. Allí también se incluye una explicación más detallada de su funcionamiento.

5. Conclusiones

Se ha diseñado y fabricado un circuito capaz de clasificar en tiempo real de forma no supervisada patrones de entrada binarios. El circuito realiza una versión modificada del algoritmo ART 1. Esta nueva versión del

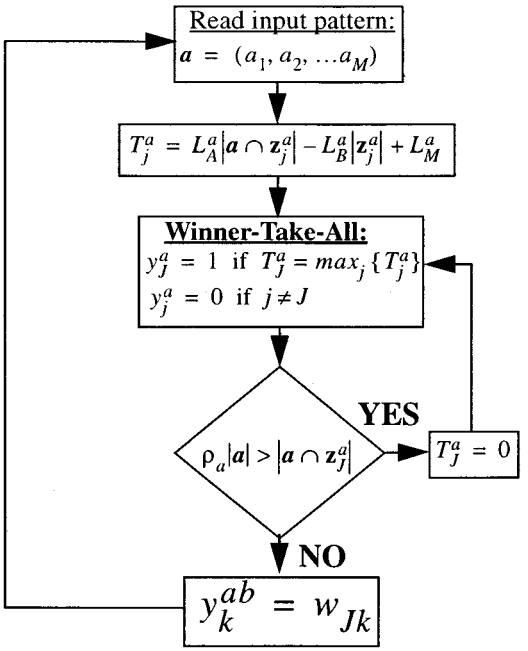


Fig. 15: Diagrama de Flujo de la Operación de Predicción del Algoritmo ARTMAP

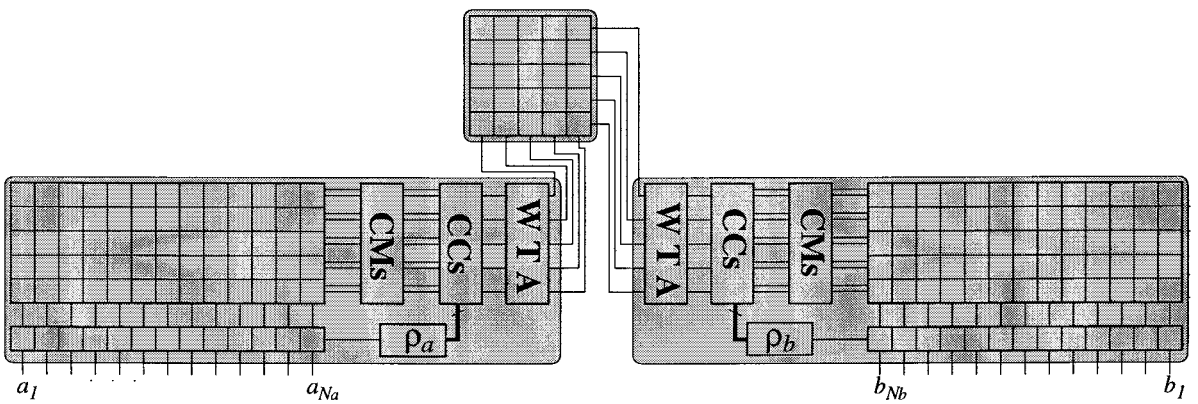


Fig. 16: Diagrama de Interconexión del Sistema ARTMAP

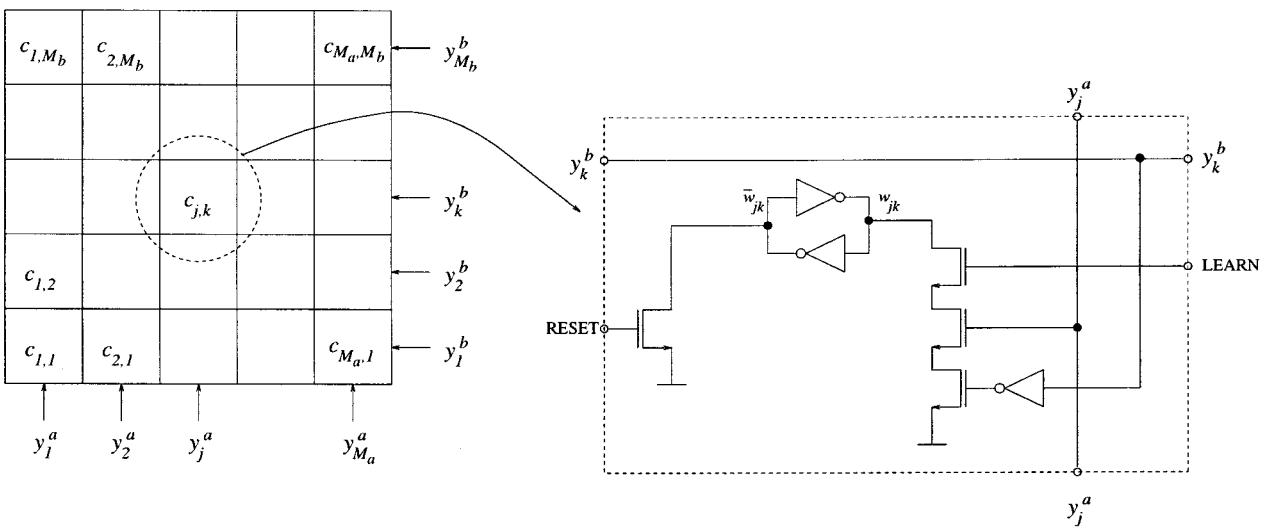


Fig. 17: Esquemático del Módulo Inter-ART

algoritmo ART 1 es mucho más eficiente para la implementación en "hardware", sin embargo, conserva todas las propiedades computacionales del algoritmo original.

Se han fabricado dos prototipos del sistema categorizador ambos en la tecnología de ES2-1.0μm. El primer prototipo clasifica patrones de 100 entradas binarias en hasta 18 categorías diferentes, ocupando un área de $1cm^2$. El segundo prototipo clasifica patrones de 50 entradas binarias en 10 categorías y ocupa un área 15 veces menor que el anterior prototipo.

Ambos prototipos son totalmente modulares, esto es, pueden ser ensamblados en matrices de $N \times M$ chips para formar sistemas mayores, con mayor número de entradas y/o mayor número de categorías. Se ha testado el comportamiento de un sistema formado por la interconexión de dos chips del segundo prototipo. Los chips estaban interconectados para aumentar el número de categorías.

Se ha construído un sistema, basado en el algoritmo ARTMAP, capaz de aprender de forma supervisada correspondencias entre pares de patrones de entrada binarios. Este sistema está formado por la interconexión de dos módulos ART 1 mediante un pequeño módulo inter-ART. El comportamiento de este sistema ha sido ampliamente testado.

Para la realización de la operación del "Winner-Take-All" característica del proceso de selección de categorías del algoritmo ART 1, se ha diseñado un circuito WTA que opera en modo de corriente. Reflejando adecuadamente las corrientes entre los distintos chips, la operación del sistema se puede distribuir entre distintos chips sin pérdida de precisión.

Se ha desarrollado un método para la caracterización estadística de las desviaciones de los parámetros eléctricos de cualquier proceso CMOS. El método requiere la realización de un chip de propósito específico en la tecnología a caracterizar. Se ha realizado el diseño del chip y las medidas para la caracterización de dos tecnologías CMOS: la tecnología de ES2-1.0μm y la del CNM-2.5μm.

6. Referencias

[1] AFCEA International Press, "Darpa Neural Network Study", A Division of the Armed Forces Communications and Electronics Association, November 1988.

[2] Proceedings of the IEEE International Joint Conference on Neural Networks, Baltimore, 1992.

[3] J. A. Anderson and E. Rosenfeld: "Neurocomputing Foundations of Research". MIT Press 1988.

[4] J. H. Chung, H. Yoon, y S. R. Maeng, "A Systolic Array Exploiting the Inherent Parallelisms of Artificial Neural Networks," *Microprocessing and Microprogramming*, vol. 33, p. 145-159, 1992.

[5] T. Kohonen, *Self-Organization and Associative Memory*, 3rd ed., Berlin, Germany: Springer-Verlag, 1989.

[6] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York: Plenum, 1981.

[7] R. Duda y P. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

[8] J. Hartigan, *Clustering Algorithms*, New York: Wiley, 1975.

[9] R. Dubes y A. Jain, *Algorithms that Cluster Data*, Englewood Cliffs, NJ: Prentice Hall, 1988.

[10] G. A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer Vision, Graphics, and Image Processing*, vol. 37, p. 54-115, 1987.

[11] G. A. Carpenter and S. Grossberg, "ART 2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns," *Applied Optics*, vol. 26, No. 23, p. 4919-4930, 1 Diciembre 1987.

[12] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System," *Neural Networks*, vol. 4, p. 759-771, 1991.

[13] G. A. Carpenter and S. Grossberg, "ART 3: Hierarchical Search Using Chemical Transmitters in Self-Organizing

Pattern Recognition Architectures," *Neural Networks*, vol. 3, p. 129-152, 1990.

[14] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network," *Neural Networks*, vol. 4, p. 565-588, 1991.

[15] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. B. Rosen, "Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps," *IEEE Transactions on Neural Networks*, vol. 3, No. 5, p. 698-712, Septiembre 1992.

[16] J. Lazzaro, R. Ryckebush, M. A. Mahowald, and C. Mead, "Winner-Take-All Networks of O(n) Complexity," in *Advances in Neural Information Processing Systems,* vol. 1, D. S. Touretzky (Ed.), Los Altos, CA: Morgan Kaufmann, 1989, p. 703-711.

[17] M. J. M. Pelgrom, A. C. J. Duinmaijer, y A. P. G. Welbers, "Matching Properties of MOS Transistors," *IEEE Journal of Solid-State Circuits,* vol. 24, No. 5, p. 1433-1440, Octubre 1989.

[18] P. E. Allen y D. R. Holberg, *CMOS Analog Circuit Design*. New York: Holt, Rinehart and Winston, Inc., 1987.

Appendix 1: A VLSI-Friendly ART 1 Algorithm

1.1. The Adaptive Resonance Theory

Since 1987 Carpenter and Grossberg have published a series of papers of **A**daptive **R**esonance **T**heory (ART) architectures [1.2], [1.3], [1.4], [1.5], [1.6] and [1.7]. These architectures are:

- ART 1 [1.2] is an architecture, which is capable of generating in an unsupervised way, stable recognition codes in response to a series of arbitrarily ordered and arbitrarily many and complex binary input patterns.
- The ART 2 [1.3] and Fuzzy-ART [1.5] architectures, which do the same but for analog input patterns.
- The ART 3 [1.4] architecture, which copes with sequences of asynchronous analog input patterns in real time.
- The ARTMAP [1.6] and Fuzzy-ARTMAP [1.7] architectures, which can be trained to learn in a supervised way correspondences between pairs of binary and analog input patterns, respectively.

The most basic module, which is the ART 1 architecture, is depicted in Fig. 1.1. It is composed of two subsystems: the "orienting subsystem" and the "attentional subsystem". The attentional subsystem consists of two layers: the F_1 or input layer and the F_2 or category layer. The F_1 layer has N cells $(v_1,v_2,...,v_N)$ and F_2 has M cells $(u_1,u_2,...,u_M)$. Each F_1 cell v_i connects to each F_2 cell u_j through bottom-up connections z_{ij}^{bu}. Each F_2 cell u_j connects to each F_1 cell through top-down connections z_{ji}^{td}. Cells in the F_2 layer have interconnections

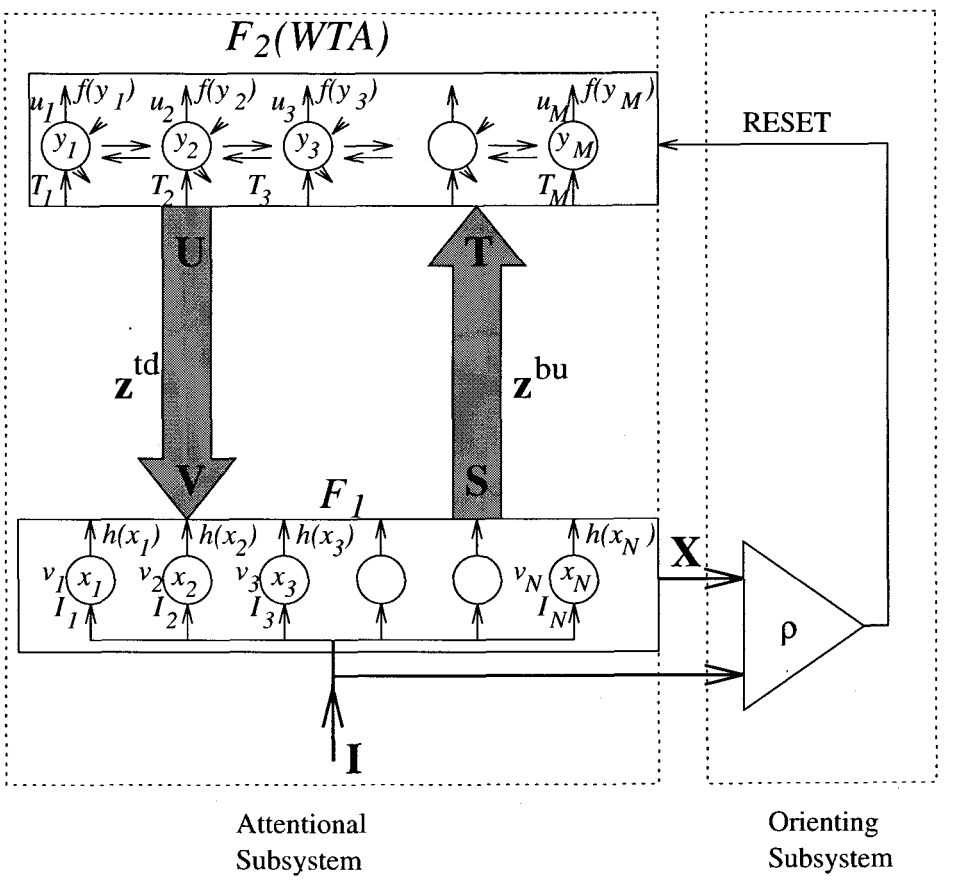


Fig. 1.1: Block Diagram of an ART 1 Architecture

This Appendix is an amplified version of paper [1.1].

among themselves, while cells in the F_1 layer are not interconnected among themselves. The activation (or state) of an F_1 cell v_i is called x_i. The activation of layer F_1 is represented by the vector $\mathbf{X} = (x_1, x_2, \ldots, x_N)$. The activation of an F_2 cell u_j is called y_j. The activation of layer F_2 is represented by the vector $\mathbf{Y} = (y_1, y_2, \ldots, y_M)$. The output of an F_1 cell v_i is a non-linear function of its activation $h(x_i)$. The output of layer F_1 is represented by the vector $\mathbf{S} = (h(x_1), h(x_2), \ldots, h(x_N))$. The output of an F_2 cell u_j is also a non-linear function of its activation $f(y_j)$, and the output of F_2 layer is represented by the vector $\mathbf{U} = (f(y_1), f(y_2), \ldots, f(y_M))$.

The ART 1 module performs the following sequence of operations:

- An input pattern $\mathbf{I} = (I_1, I_2, \ldots, I_N)$ is presented to the F_1 layer. Each pixel I_i is either 0 or 1. A pattern of activation $\mathbf{X}$ is produced across the F_1 nodes.
- The activation pattern $\mathbf{X}$ causes the so called "postsynaptic pattern of gated signals" $\mathbf{S}$ to appear at the output of the F_1 layer.
- These postsynaptic signals are multiplied by the weight matrix $\mathbf{z}^{bu}$ generating a pattern of input signals $\mathbf{T}$ at the input of the F_2 layer:

$$T_j = \sum_{i=1}^{N} z_{ij}^{bu} S_i = \sum_{i=1}^{N} z_{ij}^{bu} h(x_i) . \tag{1.1}$$

- The F_2 layer nodes interact among themselves with lateral inhibition causing a pattern of activation $\mathbf{Y}$ which is the result of contrast-enhancing the F_2 input pattern $\mathbf{T}$.
- This pattern $\mathbf{Y}$ is gated generating the F_2 output pattern $\mathbf{U}$ which is then multiplied by the weight matrix $\mathbf{z}^{td}$ to produce another input source to the F_1 layer $\mathbf{V}$ named "top-down template" or "learned expectations",

$$V_i = \sum_{j=1}^{M} z_{ji}^{td} U_j = \sum_{j=1}^{M} z_{ji}^{td} f(y_j) . \tag{1.2}$$

- This "top-down template" $\mathbf{V}$ affects the input layer F_1, causing a modification of the activation pattern across F_1. Let's call $\mathbf{X}^*$ the new activation pattern. The amount of modification which experiments the original activation pattern $\mathbf{X}$ depends on the matching between the input pattern $\mathbf{I}$ and the top-down expectations $\mathbf{V}$.
- A poor matching between $\mathbf{I}$ and $\mathbf{V}$ can cause the orienting subsystem to send a reset signal to the F_2 layer. The matching criterion used by the orienting subsystem to decide whether or not to send a reset signal is controlled by the vigilance parameter ρ. Usually, ρ ranges from 0 to 1. For values of ρ close to 1, a good matching is demanded, while for ρ values close to 0, poor matchings are acceptable. This reset signal permanently inhibits all the activated nodes across F_2, so that if an F_2 node u_j is activated the reset signal removes all its activity y_j as long as the same input pattern $\mathbf{I}$ is present at the system input.

- When the activated pattern **Y** is removed, the top-down template is also removed. So, the original F_1 activation pattern **X** and the input pattern **T** is restablished. Due to the permanent F_2 inhibition, another activation pattern $\mathbf{Y}^*$ appears across the F_2 layer, and another top-down template $\mathbf{V}^*$ is read out at the F_1 input.
- If a poor matching between **I** and $\mathbf{V}^*$ is again observed, a new reset process and a new selection process take place. This search process ends when a top-down template **V** is found which matches the input pattern **I** to the degree of accuracy required by the vigilance parameter ρ.

The modifications of the activation patterns or "short-term-memories (STM)" **X** and **Y** take place much faster than the updating of the weight matrixes or "long-term-memories (LTM)" $\mathbf{z}^{bu}$ and $\mathbf{z}^{td}$. Consequently, it can be stated that the learning process takes place when the system has established its STM state. This is often referred to as the STM being in a resonant state.

The ART 1 architecture has a collection of interesting computational properties, rarely present in other neural network systems:

- *Self-Scaling:* Features of an input pattern that are considered as irrelevant noise when embedded in a complex input pattern can be considered as critical when present in a simpler input pattern.
- *Vigilance or Variable Coarseness:* The matching criterion between the input pattern and the top-down template learned by a chosen category is adjustable, and it is determined by the vigilance parameter ρ. If the vigilance parameter is high, more attention will be paid to distinguish very similar input patterns, and classify them into different categories. However, if the vigilance parameter is low, there must be a significant difference between two input patterns to separate them into different categories.
- *Subset and Superset Direct Access:* The system is able to classify a new input pattern as belonging to either a subset or a superset category, depending on global similarity criteria. No restrictions on input orthogonality or linear predictability are needed.
- *Direct Access to Familiar Input Patterns:* No matter how many and how complex the learned patterns may be, the system always accesses directly the recognition code of a familiar previously learned input pattern.
- *Self-Stabilization:* In response to an arbitrary list of binary input patterns, all interconnection weights subject to learning approach limits after a finite number of learning trials. Learning is guaranteed to stabilize, and it does so for a small number of training pattern presentations.
- *Biasing the Network to form New Categories:* There is a parameter that can bias the tendency of the system to code unfamiliar patterns into new categories, independent of the vigilance parameter.
- *Self-Adjustment of the Search Order:* The search order is not predetermined. The whole system adjusts the search order as the self-organizing process evolves.

- *On-Line Learning:* The ART1 algorithm learns as it performs, as opposed to other algorithms, where first the algorithm must be trained and second, it can be used in an application. The ART1 algorithm can incorporate new knowledge as it is being used. This property makes ART1 an excellent candidate for real-time clustering.
- *Capturing Rare Events:* ART1 is able to identify and build clusters of events that appear with a very low frequency. Even if an event corresponding to a clearly distinct cluster appears only once, ART1 is able to detect it while building and preserving the corresponding cluster or category.

1.2. The ART 1 Mathematical Model

The ART 1 algorithm is a self-organizing system capable of learning stable recognition codes in response to a set of arbitrary many and complex binary input patterns.

The model is fully described by two sets of STM and LTM differential equations and the operation of the vigilance subsystem. The STM equations describe the evolution of the activities, x_i and y_j, in the nodes of layers F_1 and F_2. The LTM equations describe the evolution of the weights, z_{ij}^{bu} and z_{ji}^{td}, or learning rules.

As mentioned in the previous section, the STM equations evolve with a much faster time constant than that of the LTM equations.This allows us to distinguish three different levels of ART 1 implementations:

Type-1 *Full Model Implementation:* Both sets of STM and LTM time-domain differential equations are directly implemented. This implementation is the most expensive (both in hardware and in software) requiring a large amount of computational power.

Type-2 *STM Steady-State Implementation:* If an input pattern **I** is held at the F_1 input layer, until the STM equations settle to their steady state, the resulting steady state depends only on the interconnection weights, $\mathbf{z}^{bu}$ and $\mathbf{z}^{td}$, and the input pattern **I**. Consequently, it is not necessary to solve the STM differential equations. The steady state can be computed by solving the corresponding algebraic equations. In this case, a proper sequencing of STM events has to be introduced artificially and only the LTM differential equations are implemented.

Type-3 *Fast Learning Implementation:* If an input pattern **I** is held at the F_1 input layer until both STM and LTM differential equations settle to their steady state, the resulting STM and LTM steady states can be computed directly without solving any differential equation. In this case, a proper sequencing of STM and LTM events has to be done.

The full Type-1 mathematical models described next.

A. STM Equations

The STM equations which describe the evolution of activity x_i in an F_1 node v_i have the form,

$$\varepsilon \frac{dx_i}{dt} = -x_i + (1 - A_1 x_i) J_i^+ - (B_1 + C_1 x_i) J_i^- \tag{1.3}$$

where A_1, B_1 and C_1 are positive constants which assure that the x_i activity remains always limited to the interval $\left[-\frac{B_1}{C_1}, \frac{1}{A_1}\right]$. J_i^+ is the total excitatory input to F_1 node v_i and J_i^- is the total inhibitory input to F_1 node v_i.

Fig. 1.2 shows a detailed diagram of the interactions among the processing units in the system. The total excitatory input to an F_1 node v_i is the sum of the component of the input pattern I_i and the component V_i of the top-down template **V**,

$$J_i^+ = I_i + V_i = I_i + \sum_{j=1}^{M} f(y_j)\, z_{ji}^{td}. \tag{1.4}$$

The total inhibitory input to this F_1 node is the "attentional signal"

$$J_i^- = \sum_{j=1}^{M} f(y_j) \tag{1.5}$$

where $f(y_j)$ is the F_2 j-th node u_j output signal generated by its activity y_j. This attentional signal is intended to allow the system to distinguish between an input pattern **I** and a top-down template **V** generated by the

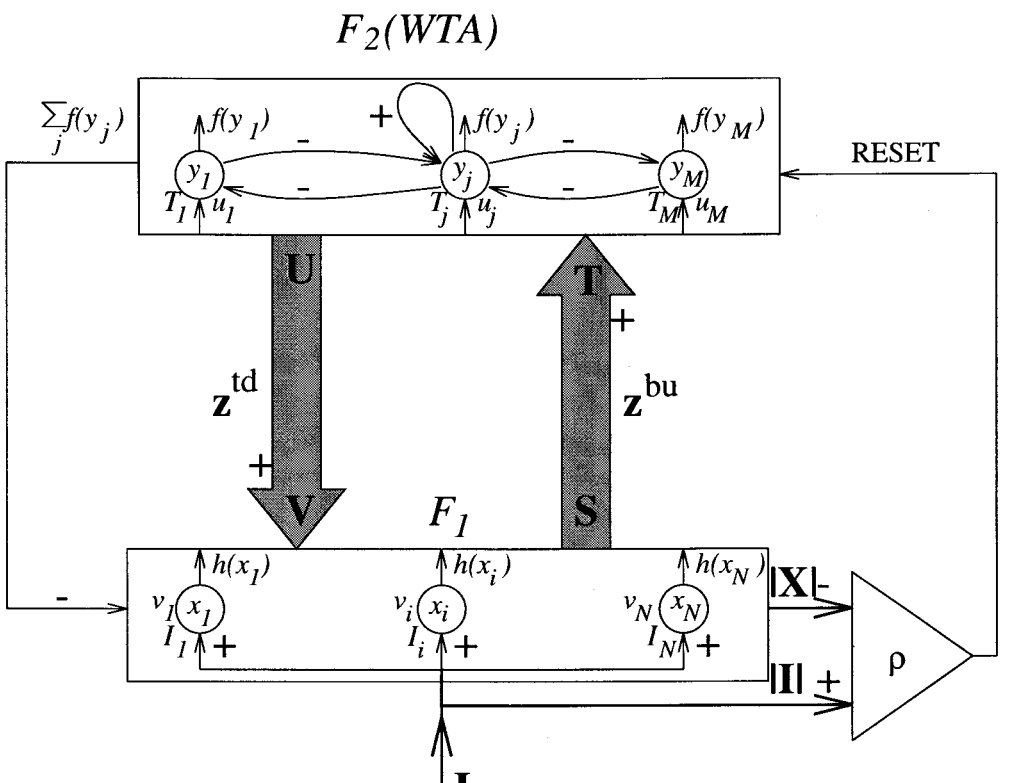


Fig. 1.2: Diagram of the interactions between the nodes in an ART 1 architecture

presence of a certain amount of activation across the layer F_2. The attentional signal $\sum_{j=1}^{M} f(y_j)$ is greater than zero whenever an activation pattern **Y** appears across layer F_2. So, if a top-down template **V** is generated before applying any input pattern **I** the attentional signal becomes active. This inhibition signal prevents the F_1 layer from becoming active without the presence of an input pattern **I**.

The same STM equation is valid to describe the evolution of the activity y_j in an F_2 node u_j,

$$\varepsilon \frac{dy_j}{dt} = -y_j + (1 - A_2 y_j) J_j^+ - (B_2 + C_2 y_j) J_j^- \tag{1.6}$$

where the amount of excitatory input equals

$$J_j^+ = T_j + f(y_j) = \sum_{i=1}^{N} z_{ij}^{bu} h(x_i) + f(y_j) \tag{1.7}$$

where $h(x_i)$ is the output signal generated by activity x_i of F_1 node v_i.

The inhibitory input is the sum of the lateral inhibitory actions from the remaining F_2 nodes, that is,

$$J_j^- = \sum_{j \neq k} f(y_k) \; . \tag{1.8}$$

Each F_2 node generates an excitatory input to itself an inhibitory inputs to the remaining F_2 nodes. This lateral inhibition contrast enhaces pattern **T** present at the F_2 layer input. If F_2 parameters are properly chosen, we can force F_2 to maintain activation only in the node u_j which receives the largest input T_j. We call this case "the forced choose situation" or "Winner-Take-All" action. In particular, if ε is small, after a reduced time,

$$f(y_j) = \begin{cases} 1 & \text{if } T_j = max\{T_k\} \\ 0 & \text{otherwise} \end{cases} \; . \tag{1.9}$$

If we denote u_J the node receiving the largest T_J input, the top-down expectation vector has the components

$$V_i = z_{Ji}^{td} \tag{1.10}$$

hence, equalling the learned top-down template of category u_J.

In this case, the total excitatory input to an F_1 node is,

$$J_i^+ = I_i + z_{Ji}^{td}, \tag{1.11}$$

and the inhibitory input J_i^- equals always '1' after a u_J node has been activated and its top-down template has been read out.

The F_1 parameters are chosen so that only the nodes receiving more excitatory than inhibitory inputs become active, so that,

$$x_i = \begin{cases} 1 \text{ if both } I_i \text{ and } z_{Ji}^{td} \text{ equal 1} \\ 0 \text{ otherwise} \end{cases}, \tag{1.12}$$

which can be expressed,

$$x_i = I_i z_{Ji}^{td}, \tag{1.13}$$

or in vector notation,

$$\mathbf{X} = \mathbf{I} \cap \mathbf{z}_J^{td}. \tag{1.14}$$

B. The LTM equations

The LTM equations define the learning rules describing the evolution of the top-down and bottom-up weight templates. The equations have the form,

$$\frac{dz_{ij}^{bu}}{dt} = f(y_j)\left[-\left((L-1) + \sum_k h(x_k)\right) z_{ij}^{bu} + Lh(x_i)\right] \tag{1.15}$$

$$\frac{dz_{ji}^{td}}{dt} = f(y_j)\left[-z_{ji}^{td} + h(x_i)\right] \tag{1.16}$$

From these equations, it can be seen that only the weight vectors $\mathbf{z}_J^{bu}$ and $\mathbf{z}_J^{td}$ corresponding to the activated F_2 node u_J evolve towards their new stationary values, which, in both cases, are a certain scaled version of the activation pattern across F_1,

$$z_{iJ}^{bu} \to \frac{Lh(x_i)}{L-1+\sum_k h(x_k)} = \frac{LI_i z_{Ji}^{td}}{L-1+\left|\mathbf{I} \cap \mathbf{z}_J^{td}\right|} \tag{1.17}$$

$$z_{Ji}^{td} \to h(x_i) = I_i z_{Ji}^{td} \tag{1.18}$$

Before any prior learning has occurred in the system, the weights are initialized to the positive values such that

$$0 < z_{ij}^{bu}(0) \le \frac{L}{L-1+N} \tag{1.19}$$

$$z_{ji}^{td}(0) = 1 \tag{1.20}$$

C. The Reset Subsystem

The reset subsystem operates on the nodes of the category layer. Whenever an unacceptable mismatch between the binary input pattern $\mathbf{I}$ and the activation pattern $\mathbf{X}$ across F_1 occurs, a reset signal is generated. This signal permanently inhibits the activated F_2 node u_J. The reset signal is activated each time the following condition is verified,[1]

$$|\mathbf{X}| < \rho|\mathbf{I}| \tag{1.21}$$

The adjustable vigilance parameter ρ controls the degree of matching required between both templates.

1.3. The Fast Learning ART 1 Algorithm

In the fast learning algorithm or *Type-3* implementation of the ART 1 model, the assumption is made that the weights reach their stationary values during each presentation of an input pattern. The flow diagram of the Type-3 ART 1 algorithm is depicted Fig. 1.3a.

The sequence of operation is as follows:

- Before the presentation of any input pattern, the weights are initialized to the values

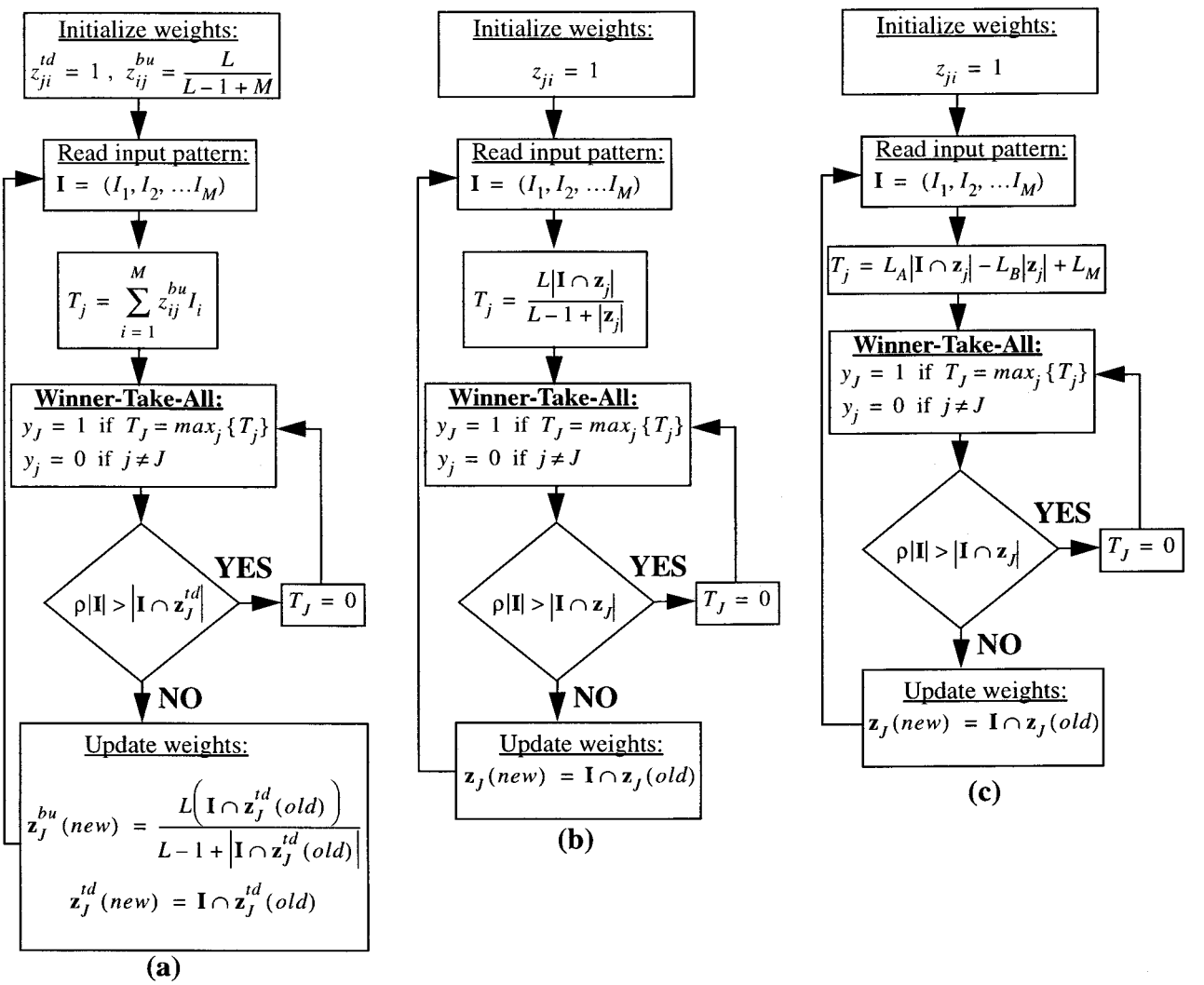


Fig. 1.3: Type-3 implementation algorithm of the ART 1 architecture: (a) original ART 1 (b) ART 1 with a single binary valued weight template (c) and VLSI-friendly ART 1_m

---

1. If **a** is a vector of components $(a_1, a_2, \ldots, a_q)$, the notation $|\mathbf{a}|$ means its L_1 norm: $|\mathbf{a}| = \sum_{i=1}^{q} |a_i|$

$$z_{ij}^{bu}(0) = \frac{L}{L-1+N} \tag{1.22}$$

$$z_{ji}^{td}(0) = 1 \tag{1.23}$$

- After an input pattern $\mathbf{I}$ is read, the T_j terms are computed

$$T_j = \sum_{i=1}^{N} z_{ij}^{bu} I_i \tag{1.24}$$

- The F_2 category u_J with the largest T_j input is selected and becomes active ($y_J = 1$), while the other F_2 nodes remain inactive ($y_{j \neq J} = 0$).
- The vigilance criterion is checked for the winning category. If $\left|\mathbf{I} \cap \mathbf{z}_J^{td}\right| < \rho|\mathbf{I}|$, the u_J category is reset and another winning category is chosen. Otherwise, the category u_J is accepted and its weights are updated to their new values given by equations (1.17) and (1.18).

1.4. The Modified ART 1 Algorithm

From a hardware implementation point of view, one of the first issues that comes into consideration is that there are two templates of weights to be built. The set of bottom-up weights z_{ij}^{bu}, each of which must store a real value belonging to the interval [0,1], and the set of top-down weights z_{ji}^{td}, each of which stores either the value '0' or '1'. The physical implementation of the bottom-up template memory presents the first hardware difficulty because the weights need either an analog or a digital memory with sufficient bits per weight so that the digital discretization does not affect the system performance. However, it can be seen from eqs.(1.17) and (1.18) that the bottom-up set $\{z_{ij}^{bu}\}$ and the top-down set $\{z_{ji}^{td}\}$ contain the same information: each of these sets can be fully computed by knowing the other set. The bottom-up set $\{z_{ij}^{bu}\}$ is a normalized version of the top-down set $\{z_{ji}^{td}\}$. Therefore, from a hardware implementation point of view, it would be desirable to implement physically only a binary valued set (one bit per weight) and introduce the normalization of the bottom-up weights during the computation of $\{T_j\}$. This way, the two sets $\{z_{ij}^{bu}\}$ and $\{z_{ji}^{td}\}$ can be substituted by a single binary valued set $\{z_{ij}\}$, and eq. (1.24) modified to take into account the normalization effect of the original bottom-up weights,[2]

$$T_j = \frac{L\left|\mathbf{I} \cap \mathbf{z}_j\right|}{L-1+\left|\mathbf{z}_j\right|} = \frac{L\sum_{i=1}^{N} z_{ij} I_i}{L-1+\sum_{i=1}^{N} z_{ij}}. \tag{1.25}$$

Considering this minor "implementation" modification, the algorithm of Fig. 1.3(a) would be transformed into that depicted in Fig. 1.3(b). The system level performance of the algorithms described by Fig. 1.3(a) and (b) is

2. This type of modification is employed in the Fuzzy-ART model [1.5], which operates with analog patterns, instead of binary ones. Making Fuzzy-ART to work with binary patterns results in ART 1 behavior, but using only one set of weights, similar to the system described in this Appendix.

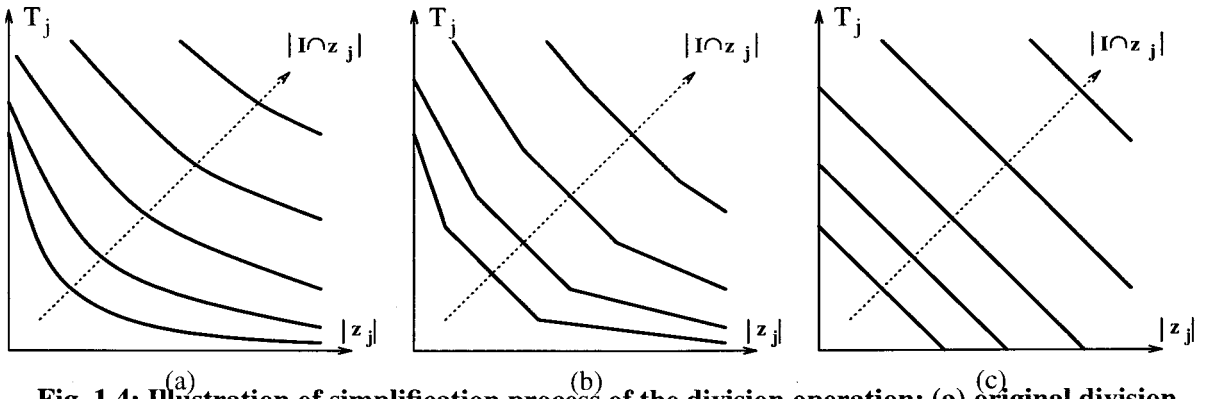


Fig. 1.4: Illustration of simplification process of the division operation: (a) original division operation, (b) piece-wise linear approximation, (c) linear approximation

identical. There is no difference in the behavior between the two diagrams, and the one in Fig. 1.3(b) offers more attractive features from a hardware (as well as software) implementation point of view.

However, in Fig. 1.3(b), an extra division operation, $T_j = (L|\mathbf{I} \cap \mathbf{z}_j|) / (L - 1 + |\mathbf{z}_j|)$, needs to be performed for each node in the F_2 layer. This is an expensive hardware operation and would probably constitute a performance bottleneck in the overall system for both analog and digital circuit implementations. If possible, it would be very desirable to avoid this division operation. That can be done by substituting this division operation by another, less expensive one, and, although this results in a system with a slightly different behavior, we will show that it preserves all the computational properties of the original ART 1 algorithm.

Fig. 1.4(a) shows the curves that represent the division operation of eq. (1.25). A first simplification could be to substitute these curves by a piece-wise linear approximation as shown in Fig. 1.4(b). Such an approximation still presents some hardware difficulties and could also limit the performance of the overall system. A more drastic simplification would be to substitute the original operation by the operation represented by the set of curves of Fig. 1.4(c). Mathematically, the division operation has been substituted by a subtraction operation[3],

$$T_j = L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M, \tag{1.26}$$

where L_A and L_B are positive parameters that play the role of the original L (and L–1) parameter. As we will see in the next Section, the condition $L_A > L_B$ must be imposed for proper system operation. $L_M > 0$ is a constant parameter needed[4] to ensure that $T_j \geq 0$ for all possible values of $|\mathbf{I} \cap \mathbf{z}_j|$ and $|\mathbf{z}_j|$.

Replacing a division operation with a subtraction one is a very important hardware simplification with significant performance improvement potential. Fig. 1.3(c) shows the final *Type-3* modified ART 1 algorithm,

---

3. Similar T_j functions (also called *distances* or *choice functions*) have been proposed by other authors for Fuzzy-ART. Since ART 1 can be considered a particular case of Fuzzy-ART when the input patterns are binary, Fuzzy-ART *choice functions* can also be used for ART 1. In section 1.8 we show how these other *choice functions* also yield to ART 1 architectures that preserve as well all the original computational properties. However, the *choice function* presented here is computationally less expensive and is easier to implement in hardware.
4. In reality, parameter L_M has been introduced for hardware reasons [1.13] and [1.14]. In a software implementation parameter L_M can be ignored.

which we will call from now on the ART 1_m algorithm. In the next sections, we will try to show that the price paid for this drastic simplification, although it yields a system with slightly different input-output behavior, is insignificant since all the computational properties of the original ART 1 architecture are preserved.

It is worth mentioning here that substituting a division operation by a subtraction one means a significant performance boost from a hardware implementation point of view. Implementing physically division operators in hardware constraints significantly the whole system design and imposes limitations on the overall system performance.

In the case of digital hardware, a division circuit can be built using either sequential techniques or large size higher speed special purpose circuits [1.10]. Sequential techniques use simpler hardware but are slower, while a dedicated circuit is very large compared to the former and requires much more power consumption. As an example, and for a sequential type division circuit, in order to realize the following division

$$T_j = \frac{L|\mathbf{I} \cap \mathbf{z}_j|}{L-1+|\mathbf{z}_j|}, \tag{1.27}$$

q addition/substractions operation would be needed, where q is the number of bits needed for the result of the division. If, for example, there are $N = 1000$ nodes in the F_1 layer, numerator and denominator in eq. (1.27) should be represented by 10-bit words. If, for a given input $\mathbf{I}$, we want to differentiate between two terms T_{j_1} and T_{j_2} whose respective templates $\mathbf{z}_{j_1}$ and $\mathbf{z}_{j_2}$ differ in one bit, the F_2 layer (WTA) would need to resolve

$$|\Delta T_{j_1 j_2}|_{min} = \left| \frac{L|\mathbf{I} \cap \mathbf{z}_{j_1}|}{L-1+|\mathbf{z}_{j_1}|} - \frac{L|\mathbf{I} \cap \mathbf{z}_{j_2}|}{L-1+|\mathbf{z}_{j_2}|} \right|_{min}. \tag{1.28}$$

The worst case occurs when $|\mathbf{z}_{j_1}| = |\mathbf{I} \cap \mathbf{z}_{j_1}| = N$, $|\mathbf{z}_{j_2}| = |\mathbf{I} \cap \mathbf{z}_{j_2}| = N-1$. In this case

$$|\Delta T_{j_1 j_2}|_{min} = \left| \frac{L(L-1)}{(L-1+N)(L-2+N)} \right|. \tag{1.29}$$

A reasonable minimum value for L is 1.01. Therefore, if $N = 1000$ then $|\Delta T_{j_1 j_2}|_{min} \approx 10^{-8}$. On the other hand, it is easy to see that $|\Delta T_{j_1 j_2}|_{max}$ is close to but less than one. Consequently, for each T_j a dynamic range of

$$\frac{|T_j|_{max}}{|T_j|_{min}} = 10^8 \tag{1.30}$$

is needed. Such dynamic range requires a q=27 bit representation. Thus, for each division operation we need to realize 27 10-bit addition/subtractions. Furthermore, the WTA in the F_2 layer would need to choose the maximum among M 27-bit words. On the other hand, if the ART 1_m algorithm is used, instead of the $M \times 24$ 11-bit addition/subtractions, we need only to realize M 11-bit subtractions, and the WTA has to choose the maximum among M 11-bit words.

In the case of analog hardware, there are ways to implement the division operation with compact dedicated circuits [1.9], [1.12], [1.11], [1.15], but they usually suffer from low signal-to-noise ratios, limited signal range, noticeable distortion, or require bipolar devices which are available for more expensive VLSI technologies. In any case, the performance of the overall ART system would be limited by the lower

performance of the division operators. If the division operators are eliminated the performance of the system would be limited by other operators which, for the same VLSI technology, render considerable better performance figures. Furthermore, in the case of analog current mode signal processing [1.14], the addition and subtraction of currents does not need any physical components. Consequently, by eliminating the need of signal division, the circuitry is dramatically simplified and its performance drastically improved.

1.5. Computational Equivalence of the Original and the Modified Models

Throughout the original ART 1 paper [1.2], Carpenter and Grossberg provide rigorous demonstrations of the computational properties of the ART 1 architecture. Some of these properties are concerned with *Type-1* and *Type-2* operations of the architecture, but most refer to the *Type-3* model operation. From a functional point of view, i.e., when looking at the ART 1 system as a black box regardless of the details of its internal operations, the system level computational properties of ART 1 are fully contained in its *Fast-Learning* or *Type-3* model. The theorems and demonstrations given by Carpenter and Grossberg [1.2] relating to *Type-1* and *Type-2* models of the system only ensure proper *Type-3* behavior. The purpose of this Section is to demonstrate that the modified *Type-3* model developed during the previous Section preserves all the *Type-3* computational properties of the original ART 1 architecture. The only functional difference between ART 1 and ART 1_m, is the way the terms T_j are computed before competing in the Winner-Take-All block. Therefore, the original properties and demonstrations that are not affected by the terms T_j will be automatically preserved. Such properties are, for example, the *Self-Scaling* property and the *Variable Coarseness* property tuned by the *Vigilance Parameter.* But there are other properties which are directly affected by the way the terms T_j are computed. In the remainder of this Section we will show that these properties remain in the ART 1_m architecture.

Let us define a few concepts before demonstrating that the original computational properties are preserved.

a) *Direct Access:* an input pattern **I** is said to have *Direct Access* to a learned category u_j if this category is the first one selected by the Winner-Take-All F_2 layer and is accepted by the *vigilance subsystem*, so that no reset occurs.

b) *Subset Template:* an input pattern **I** is said to be a *Subset Template* of a learned category $\mathbf{z}_j \equiv (z_{1j}, z_{2j}, \ldots z_{Nj})$ if $\mathbf{I} \subset \mathbf{z}_j$. Formally,

$$\begin{aligned} &z_{ij} = 0 \Rightarrow I_i = 0 \qquad \forall i = 1, \ldots N, \\ &I_i = 1 \Rightarrow z_{ij} = 1 \qquad \forall i = 1, \ldots N, \\ &\text{there are some values of } i \text{ such that} \quad I_i = 0 \quad \text{and} \quad z_{ij} = 1. \end{aligned} \tag{1.31}$$

c) *Superset Template:* an input pattern **I** is said to be a *Superset Template* of a learned category u_j if $\mathbf{z}_j \subset \mathbf{I}$.

d) *Mixed Template:* $\mathbf{z}_j$ and $\mathbf{I}$ are said to be mixed templates if neither $\mathbf{I} \subset \mathbf{z}_j$ nor $\mathbf{z}_j \subset \mathbf{I}$ are satisfied, and $\mathbf{I} \neq \mathbf{z}_j$.

e) *Uncommitted node:* an F_2 node u_j is said to be uncommitted if all its weights z_{ij} ($i = 1, \ldots N$) preserve their initial value ($z_{ij} = 1$), i.e., node u_j has not yet been selected to represent any learned category.

A. Direct Access to Subset and Superset Patterns

Suppose that a learning process has produced a set of categories in the F_2 layer. Each category u_j is characterized by the set of weights that connect node u_j in the F_2 layer to all nodes in the F_1 layer, i.e., $\mathbf{z}_j \equiv (z_{1j}, z_{2j}, \ldots, z_{Nj})$. Suppose that two of these categories, u_{j_1} and u_{j_2}, are such that $\mathbf{z}_{j_1} \subset \mathbf{z}_{j_2}$ ($\mathbf{z}_{j_1}$ is a subset template of $\mathbf{z}_{j_2}$). Now consider two input patterns $\mathbf{I}^{(1)}$ and $\mathbf{I}^{(2)}$ such that,

$$\begin{aligned} \mathbf{I}^{(1)} &= \mathbf{z}_{j_1} \equiv (z_{1j_1}, z_{2j_1}, \ldots z_{Nj_1}) \quad , \\ \mathbf{I}^{(2)} &= \mathbf{z}_{j_2} \equiv (z_{1j_2}, z_{2j_2}, \ldots z_{Nj_2}) \quad . \end{aligned} \tag{1.32}$$

The *Direct Access to Subset and Superset* property assures that input $\mathbf{I}^{(1)}$ will have *Direct Access* to category u_{j_1} and that input $\mathbf{I}^{(2)}$ will have *Direct Access* to category u_{j_2}.

A.1 : Original ART 1:

Let us compute the values of T_{j_1} and T_{j_2} when the input patterns $\mathbf{I}^{(1)}$ and $\mathbf{I}^{(2)}$ are presented at the input of the system. For pattern $\mathbf{I}^{(1)}$ we will have,

$$\begin{aligned} T_{j_1} &= \frac{L\sum_{i=1}^{N} I_i^{(1)} z_{ij_1}}{L-1+|\mathbf{z}_{j_1}|} = \frac{L|\mathbf{I}^{(1)}|}{L-1+|\mathbf{I}^{(1)}|} \\ T_{j_2} &= \frac{L\sum_{i=1}^{N} I_i^{(1)} z_{ij_2}}{L-1+|\mathbf{z}_{j_2}|} = \frac{L|\mathbf{I}^{(1)}|}{L-1+|\mathbf{I}^{(2)}|} \end{aligned} \tag{1.33}$$

Since $|\mathbf{I}^{(1)}| < |\mathbf{I}^{(2)}|$, it is obvious that $T_{j_1} > T_{j_2}$ (remember that $L>1$) and therefore category u_{j_1} will become the active one. On the other hand, if input pattern $\mathbf{I}^{(2)}$ is presented at the input,

$$\begin{aligned} T_{j_1} &= \frac{L\sum_{i=1}^{N} I_i^{(2)} z_{ij_1}}{L-1+|\mathbf{z}_{j_1}|} = \frac{L|\mathbf{I}^{(1)}|}{L-1+|\mathbf{I}^{(1)}|} \\ T_{j_2} &= \frac{L\sum_{i=1}^{N} I_i^{(2)} z_{ij_2}}{L-1+|\mathbf{z}_{j_2}|} = \frac{L|\mathbf{I}^{(2)}|}{L-1+|\mathbf{I}^{(2)}|} \end{aligned} \tag{1.34}$$

Since the function *Lx/(L-1+x)* is an increasing function of *x*, it results that now $T_{j_2} > T_{j_1}$ and category u_{j_2} will be chosen as the winner.

A.2 : Modified ART 1:

If pattern $\mathbf{I}^{(1)}$ is given as the input pattern we will have

$$\begin{aligned} T_{j_1} &= L_A \sum_{i=1}^{N} I_i^{(1)} z_{ij_1} - L_B |\mathbf{z}_{j_1}| + L_M = L_A |\mathbf{I}^{(1)}| - L_B |\mathbf{I}^{(1)}| + L_M \\ T_{j_2} &= L_A \sum_{i=1}^{N} I_i^{(1)} z_{ij_2} - L_B |\mathbf{z}_{j_2}| + L_M = L_A |\mathbf{I}^{(1)}| - L_B |\mathbf{I}^{(2)}| + L_M \end{aligned} \tag{1.35}$$

Since $|\mathbf{I}^{(1)}| < |\mathbf{I}^{(2)}|$, it follows that (remember that L_B>0) $T_{j_1} > T_{j_2}$. In the case pattern $\mathbf{I}^{(2)}$ is presented at the input of the network it would be,

$$\begin{aligned} T_{j_1} &= L_A \sum_{i=1}^{N} I_i^{(2)} z_{ij_1} - L_B |\mathbf{z}_{j_1}| + L_M = L_A |\mathbf{I}^{(1)}| - L_B |\mathbf{I}^{(1)}| + L_M \\ T_{j_2} &= L_A \sum_{i=1}^{N} I_i^{(2)} z_{ij_2} - L_B |\mathbf{z}_{j_2}| + L_M = L_A |\mathbf{I}^{(2)}| - L_B |\mathbf{I}^{(2)}| + L_M \end{aligned} \tag{1.36}$$

In order to guarantee that $T_{j_2} > T_{j_1}$ the condition

$$L_A > L_B \tag{1.37}$$

has to be assured.

B. Direct Access by Perfectly Learned Patterns (Theorem 1 of original ART 1):

This theorem, adapted to a *Type-3* implementation, states the following

> An input pattern $\mathbf{I}$ has direct access to a node u_J which has perfectly learned the input pattern $\mathbf{I}$.

B.1 : Original ART 1:

In order to prove that **I** has direct access to u_J, we need to demonstrate that the following properties hold: (i) u_J is the first node to be chosen, (ii) u_J is accepted by the vigilance subsystem and (iii) u_J remains active as learning takes place.

To prove property (i) we have to show that, at the start of each trial $T_J > T_j \ \forall j \neq J$. Since $\mathbf{I} = \mathbf{z}_J$,

$$T_J = \frac{L|\mathbf{I}|}{L-1+|\mathbf{I}|} \tag{1.38}$$

and

$$T_j = \frac{L|\mathbf{I} \cap \mathbf{z}_j|}{L-1+|\mathbf{z}_j|} . \tag{1.39}$$

Since $\frac{Lw}{L-1+w}$ is an increasing function of w (because $L > 1$) and $|\mathbf{I}| > |\mathbf{I} \cap \mathbf{z}_j|$, we can state,

$$T_J = \frac{L|\mathbf{I}|}{L-1+|\mathbf{I}|} > \frac{L|\mathbf{I} \cap \mathbf{z}_j|}{L-1+|\mathbf{I} \cap \mathbf{z}_j|} > \frac{L|\mathbf{I} \cap \mathbf{z}_j|}{L-1+|\mathbf{z}_j|} = T_j . \tag{1.40}$$

So, property (i) is always fulfilled.

Property (ii) is directly verified since $|\mathbf{I} \cap \mathbf{z}_j| = |\mathbf{I}| \geq \rho|\mathbf{I}| \ \ \forall \rho \in [0, 1]$.

Property (iii) is always verified because after node u_J is selected as the winning category, its weight template $\mathbf{z}_J$ will remain unchanged (because $\mathbf{z}_J|_{new} = \mathbf{I} \cap \mathbf{z}_J|_{old} = \mathbf{I} = \mathbf{z}_J|_{old}$), and consequently the inputs to the F_2 layer T_j will remain unchanged.

B.2 : Modified ART 1:

In order to demonstrate that $\mathbf{I}$ has direct access to u_J, we have only to prove that property (i) is verified for the modified algorithm, as the proof of properties (ii) and (iii) is identical to the case of the original algorithm.

To prove property (i), we have to demonstrate that

$$T_J = L_A|\mathbf{I}| - L_B|\mathbf{I}| + L_M > L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M = T_j . \tag{1.41}$$

Since $L_A w - L_B w + L_M$ is an increasing function of w ($L_A > L_B$), and $|\mathbf{z}_j| > |\mathbf{I} \cap \mathbf{z}_j|$,

$$T_J = L_A|\mathbf{I}| - L_B|\mathbf{I}| + L_M > L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{I} \cap \mathbf{z}_j| + L_M > L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M = T_j \tag{1.42}$$

C. Stable Choices in STM (Theorem 2 of original ART 1):

Whenever an input pattern $\mathbf{I}$ is presented for the first time to the ART 1 system, a set of $\{T_j\}$ values is formed that compete in the Winner-Take-All F_2 layer. The winner may be reset by the *vigilance subsystem*, and a new winner appears that may also be reset, and so on until a final winner is accepted. During this search process, the T_j values that led to earlier winners are set to zero. Let us call O_j the values of T_j at the beginning of the search process, i.e., before any of them is set to zero by the vigilance subsystem. Theorem 2 of the original ART 1 architecture states:

> Suppose that an F_2 node u_J is chosen for STM storage instead of another node u_j because $O_J > O_j$. Then read-out of the top-down template preserves the inequality $T_J > T_j$ and thus confirms the choice of u_J by the bottom-up filter.

This theorem has only sense for a *Type-1* implementation, because there, as a node in the F_2 layer activates, the initial values of T_j (immediately after presenting an input pattern $\mathbf{I}$) may be altered through the top-down

"*feed-back*" connections. In a *Type-3* description (see Fig. 1.3) the initial terms T_j remain unchanged, independently of what happens in the F_2 layer. Therefore, this theorem is implicitly satisfied.

D. Initial Filter Values determine Search Order (Theorem 3 of original ART 1):

Theorem 3 of the original ART 1 architecture states that (page 92 of [1.2]):

> The Order Function $(O_{j_1} > O_{j_2} > O_{j_3} > \ldots)$ determines the order of search no matter how many times F_2 is reset during a trial.

The proof is the same for the original ART 1 and the modified ART 1 (both *Type-3*) implementation[5]. If T_{j_1} is reset by the *vigilance subsystem*, the values of $T_{j_2}, T_{j_3}, \ldots$ will not change. Therefore, the new order sequence is $O_{j_2} > O_{j_3} > \ldots$ and the original second largest value O_{j_2} will be selected as the winner. If T_{j_2} is now set to zero, O_{j_3} is the next winner, and so on.

This Theorem, although trivial in a *Type-3* implementation, has more importance in a *Type-1* description where the process of selecting and shutting down a winner has the consequence of altering all T_j values.

E. Learning on a Single Trial (Theorem 4 of original ART 1):

This theorem (page 93 of [1.2]) states the following, assuming a *Type-3* implementation is being considered[6]:

> Suppose that an F_2 winning node u_J is accepted by the vigilance subsystem. Then the LTM traces z_{ij} change in such a way that T_J increases and all other T_j remain constant, thereby confirming the choice of u_J. In addition, the set $\mathbf{I} \cap \mathbf{z}_J$ remains constant during learning, so that learning does not trigger reset of u_J by the vigilance subsystem.

E.1 : Original ART 1:

According to eq. (1.25), if u_J is the winning category accepted by the vigilance subsystem, we have that

$$T_J = \frac{LT_{AJ}}{L - 1 + T_{BJ}} = \frac{L|\mathbf{I} \cap \mathbf{z}_J|}{L - 1 + |\mathbf{z}_J|} \quad . \tag{1.43}$$

This is the T_J value before learning takes place. After updating the weights (see Fig. 1.3(a)),

$$\mathbf{z}_J(new) = \mathbf{I} \cap \mathbf{z}_J(old) \tag{1.44}$$

and the new T_J value is given by,

---

5. However, note that the resulting ordering $\{j_1, j_2, j_3, \ldots\}$ can be different for the original and for the modified architecture.
6. In the original ART 1 paper [1.2] a more sophisticated demonstration for this theorem is provided. The reason is that there the demonstration is performed for a *Type-1* description of ART 1.

$$T_j(new) = \frac{L|\mathbf{I} \cap \mathbf{z}_J(new)|}{L-1+|\mathbf{z}_J(new)|} = \frac{L|\mathbf{I} \cap \mathbf{I} \cap \mathbf{z}_J(old)|}{L-1+|\mathbf{I} \cap \mathbf{z}_J(old)|} \geq \frac{L|\mathbf{I} \cap \mathbf{z}_J(old)|}{L-1+|\mathbf{z}_J(old)|} = T_j(old) \tag{1.45}$$

Note that by eq. (1.44),

$$\mathbf{I} \cap \mathbf{z}_J(new) = \mathbf{I} \cap \mathbf{I} \cap \mathbf{z}_J(old) = \mathbf{I} \cap \mathbf{z}_J(old) \tag{1.46}$$

Since the only weights that are updated are those connected to the winning (and accepted) u_J node, all other $T_j|_{j \neq J}$ values remain unchanged. Therefore, it can be concluded, by eq. (1.45), that learning confirms the choice of u_J and that, by eq. (1.46), the set $\mathbf{I} \cap \mathbf{z}_J$ remains constant.

E.2 : Modified ART 1:

In this case, if u_J is the winning category accepted by the *vigilance subsystem*, by eq. (1.26) we have that

$$T_J = L_A T_{AJ} - L_B T_{BJ} + L_M = L_A|\mathbf{I} \cap \mathbf{z}_J| - L_B|\mathbf{z}_J| + L_M \tag{1.47}$$

The update rule is the same as before (see Fig. 1.3(b)), therefore

$$\mathbf{z}_J(new) = \mathbf{I} \cap \mathbf{z}_J(old) \tag{1.48}$$

and the new T_J value is given now by,

$$T_J(new) = L_A|\mathbf{I} \cap \mathbf{z}_J(old)| - L_B|\mathbf{I} \cap \mathbf{z}_J(old)| + L_M \geq L_A|\mathbf{I} \cap \mathbf{z}_J(old)| - L_B|\mathbf{z}_J(old)| + L_M = T_J(old) \tag{1.49}$$

Like before, learning confirms the choice of u_J, and by eq. (1.18) the set $\mathbf{I} \cap \mathbf{z}_J$ remains constant as well.

F. Stable Category Learning (Theorem 5 of original ART 1):

Suppose an arbitrary list (finite or infinite) of binary input patterns is presented to an ART 1 system. Each template set $\mathbf{z}_j \equiv (z_{1j}, z_{2j}, \ldots z_{Nj})$ is updated every time category u_j is selected by the Winner-Take-All F_2 layer and accepted by the vigilance subsystem. Some of these times template $\mathbf{z}_j$ might be changed, and some others it might stay unchanged. Let us call the times $\mathbf{z}_j$ suffers a change $t_1^{(j)} < t_2^{(j)} < \ldots < t_{r_j}^{(j)}$. Since vector (or template) $\mathbf{z}_j$ has N components (initially set to '1'), and by eqs. (1.44) and (1.18), each component can only change from '1' to '0' but not from '0' to '1', it follows that template $\mathbf{z}_j$ can, at the most, suffer N–1 changes[7],

$$r_j \leq N-1 \tag{1.50}$$

Since template $\mathbf{z}_j$ will remain unchanged after time $t_{r_j}^{(j)}$, it is concluded that the complete LTM memory will suffer no change after time

$$t_{learn} = max_j\{t_{r_j}^{(j)}\} \tag{1.51}$$

---

7. Here we are assuming that the empty template ($z_{ij} = 0$, $\forall i$) is not a valid one. The only way to store this template is by using the empty input pattern ($I_i = 0$, $\forall i$), which we assume has no significance, and hence will never be presented.

If there is a finite number of nodes in the F_2 layer t_{learn} has a finite value, and thus learning completes after a finite number of time steps.

All this is true for both, the original and the modified ART 1 architecture, and therefore the following theorem (page 95 of [1.2]) is valid for the two algorithms:

> In response to an arbitrary list of binary input patterns, all LTM traces $z_{ij}(t)$ approach limits after a finite number of learning trials. Each template set $\mathbf{z}_j$ remains constant except for at most $N-1$ times $t_1^{(1)} < t_2^{(2)} < \ldots < t_{r_j}^{(j)}$ at which it progressively loses elements, leading to the
>
> Subset Recoding Property: $\mathbf{z}_j\left(t_1^{(j)}\right) \supset \mathbf{z}_j\left(t_2^{(j)}\right) \supset \ldots \supset \mathbf{z}_j\left(t_{r_j}^{(j)}\right)$. (1.52)
>
> The LTM traces $z_{ij}(t)$ such that $i \notin \mathbf{z}_j\left(t_1^{(j)}\right)$ decrease to zero. The LTM traces $z_{ij}(t)$ such that $i \in \mathbf{z}_j\left(t_{r_j}^{(j)}\right)$ remain always at '1'. The LTM traces such that $i \in \mathbf{z}_j\left(t_{r_k}^{(j)}\right)$ but $i \notin \mathbf{z}_j\left(t_{r_{k+1}}^{(j)}\right)$ stay at '1' for times $t \le t_k^{(1)}$ but will change to and stay at '0' for times $t \ge t_{k+1}^{(j)}$.

G. Direct Access after Learning Self-Stabilizes (Theorem 6 of original ART 1):

Assuming F_2 has a finite number of nodes, the present theorem (page 98 of [1.2]) states the following:

> After recognition learning has self-stabilized in response to an arbitrary list of binary input patterns, each input pattern $\mathbf{I}$ either has direct access to the node u_j which possesses the largest subset template with respect to $\mathbf{I}$, or $\mathbf{I}$ cannot be coded by any node of F_2. In the latter case, F_2 contains no uncommitted nodes.

Since learning has already stabilized $\mathbf{I}$ can be coded only by a node u_j whose template $\mathbf{z}_j$ is a subset template with respect to $\mathbf{I}$. Otherwise, after u_j becomes active, the set $\mathbf{z}_j$ would contract to $\mathbf{z}_j \cap \mathbf{I}$, thereby contradicting the hypothesis that learning has already stabilized. Thus if $\mathbf{I}$ activates any node other than one with a subset template, that node must be reset by the *vigilance subsystem*. For the remainder of the proof, let u_J be the first F_2 node activated by $\mathbf{I}$. We need to show that if $\mathbf{z}_J$ is a subset template, then it is the subset template with the largest O_J; and if it is not a subset template, then all subset templates activated on that trial will be reset by the vigilance subsystem. To proof these two steps we need to differentiate between the original ART 1 and the modified one.

G.1 : Original ART 1:

If u_J and u_j are nodes with subset templates with respect to $\mathbf{I}$, then

$$O_j = \frac{L|\mathbf{z}_j|}{L-1+|\mathbf{z}_j|} < O_J = \frac{L|\mathbf{z}_J|}{L-1+|\mathbf{z}_J|} \tag{1.53}$$

Since $L|\mathbf{z}_j|/(L-1+|\mathbf{z}_j|)$ is an increasing function of $|\mathbf{z}_j|$,

$$|\mathbf{z}_j| < |\mathbf{z}_J| \tag{1.54}$$

and,

$$R_j = \frac{|\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{I}|} = \frac{|\mathbf{z}_j|}{|\mathbf{I}|} < R_J = \frac{|\mathbf{I} \cap \mathbf{z}_J|}{|\mathbf{I}|} = \frac{|\mathbf{z}_J|}{|\mathbf{I}|} \tag{1.55}$$

Once activated, a node u_k will be reset if $R_k < \rho$. Therefore, if u_J is reset ($R_J < \rho$), then all other nodes with subset templates will be reset as well ($R_j < \rho$).

Now suppose that u_J, the first activated node, does not have a subset template with respect to $\mathbf{I}$ ($|\mathbf{I} \cap \mathbf{z}_J| < |\mathbf{z}_J|$), but that another node u_j with a subset template is activated in the course of search. We need to show that $|\mathbf{I} \cap \mathbf{z}_j| = |\mathbf{z}_j| < \rho|\mathbf{I}|$, so that u_j is reset. We know that,

$$O_j = \frac{L|\mathbf{z}_j|}{L-1+|\mathbf{z}_j|} < O_J = \frac{L|\mathbf{I} \cap \mathbf{z}_J|}{L-1+|\mathbf{z}_J|} < \frac{L|\mathbf{z}_J|}{L-1+|\mathbf{z}_J|} \tag{1.56}$$

which implies that $|\mathbf{z}_j| < |\mathbf{z}_J|$. Since u_J cannot be chosen, it has to be reset by the *vigilance subsystem*, which means that $|\mathbf{I} \cap \mathbf{z}_J| < \rho|\mathbf{I}|$. Therefore,

$$\frac{|\mathbf{z}_j|}{L-1+|\mathbf{z}_j|} < \frac{|\mathbf{I} \cap \mathbf{z}_J|}{L-1+|\mathbf{z}_J|} < \frac{\rho|\mathbf{I}|}{L-1+|\mathbf{z}_J|} < \frac{\rho|\mathbf{I}|}{L-1+|\mathbf{z}_j|} \tag{1.57}$$

which implies that,

$$|\mathbf{I} \cap \mathbf{z}_j| = |\mathbf{z}_j| < \rho|\mathbf{I}| \tag{1.58}$$

G.2 : Modified ART 1:

If u_J and u_j are nodes with subset templates with respect to $\mathbf{I}$, then

$$O_j = L_A|\mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M < O_J = L_A|\mathbf{z}_J| - L_B|\mathbf{z}_J| + L_M \tag{1.59}$$

Since $(L_A - L_B)|\mathbf{z}_j|$ is an increasing function of $|\mathbf{z}_j|$,

$$|\mathbf{z}_j| < |\mathbf{z}_J| \tag{1.60}$$

and,

$$R_j = \frac{|\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{I}|} = \frac{|\mathbf{z}_j|}{|\mathbf{I}|} < R_J = \frac{|\mathbf{I} \cap \mathbf{z}_J|}{|\mathbf{I}|} = \frac{|\mathbf{z}_J|}{|\mathbf{I}|} \tag{1.61}$$

Therefore, if u_J is reset ($R_J < \rho$), then all other nodes with subset templates will be reset as well ($R_j < \rho$).

Now suppose that u_J, the first activated node, does not have a subset template with respect to $\mathbf{I}$ ($|\mathbf{I} \cap \mathbf{z}_J| < |\mathbf{z}_J|$), but that another node u_j with a subset template is activated in the course of search. We need to show that $|\mathbf{I} \cap \mathbf{z}_j| = |\mathbf{z}_j| < \rho|\mathbf{I}|$, so that u_j is reset. We know that,

$$O_j = (L_A - L_B)|\mathbf{z}_j| + L_M < O_J = L_A|\mathbf{I} \cap \mathbf{z}_J| - L_B|\mathbf{z}_J| + L_M < (L_A - L_B)|\mathbf{z}_J| + L_M \tag{1.62}$$

which implies that $|\mathbf{z}_j| < |\mathbf{z}_J|$. Since u_J cannot be chosen, it has to be reset by the *vigilance subsystem*, which means that $|\mathbf{I} \cap \mathbf{z}_J| < \rho|\mathbf{I}|$. Therefore,

$$L_A|\mathbf{z}_j| - L_B|\mathbf{z}_j| < L_A|\mathbf{I} \cap \mathbf{z}_J| - L_B|\mathbf{z}_J| < L_A\rho|\mathbf{I}| - L_B|\mathbf{z}_J| < L_A\rho|\mathbf{I}| - L_B|\mathbf{z}_j| \tag{1.63}$$

which implies that,

$$|\mathbf{I} \cap \mathbf{z}_j| = |\mathbf{z}_j| < \rho|\mathbf{I}| \tag{1.64}$$

H. Search Order(Theorem 7 of original ART 1):

The original Theorem 7 (page 100 of [1.2]) states the following:

Suppose that input pattern satisfies

$$L - 1 \le \frac{1}{|\mathbf{I}|} \tag{1.65}$$

and

$$|\mathbf{I}| \le N - 1 \tag{1.66}$$

Then F_2 nodes are searched in the following order, if they are searched at all.

Subset templates with respect to $\mathbf{I}$ are searched first, in order of decreasing size. If the largest subset template is reset, then all subset templates are reset. If all subset templates have been reset and if no other learned templates exist, then the first uncommitted node to be activated will code $\mathbf{I}$. If all subset templates are searched and if there exist learned superset templates but no mixed templates, then the node with the smallest superset template will be activated next and will code $\mathbf{I}$. If all subset templates are searched and if both superset templates $\mathbf{z}_J$ and mixed templates $\mathbf{z}_j$ exist, then u_j will be searched before u_J if and only if

$$|\mathbf{z}_j| < |\mathbf{z}_J| \qquad \textit{and} \qquad \frac{|\mathbf{I}|}{|\mathbf{z}_J|} < \frac{|\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{z}_j|} \ . \tag{1.67}$$

If all subset templates are searched and if there exist mixed templates but no superset templates, then a node u_j with a mixed template will be searched before an uncommitted node u_J if and only if

$$\frac{L|\mathbf{I} \cap \mathbf{z}_j|}{L - 1 + |\mathbf{z}_j|} > T_J(\mathbf{I}, \text{t=0}) \ . \tag{1.68}$$

Where $T_J(\mathbf{I}, \text{t=0}) = (L\sum I_i z_{iJ}(0)) / (L - 1 + \sum z_{iJ}(0))$. The conditions expressed in eqs. (1.65)-(1.68) have to be changed in order to adapt this theorem to the modified ART 1 architecture. The original proof will not be reproduced here, because it differs drastically from the one we will provide for the modified theorem. The modified theorem is identical to the original one, except for eqs. (1.65)-(1.68). It states the following:

Suppose that input pattern satisfies

$$\frac{L_A}{L_B} < \frac{N}{N - 1} \tag{1.69}$$

and

$$|\mathbf{I}| \le N - 1 \tag{1.70}$$

Then F_2 nodes are searched in the following order, if they are searched at all.

Subset templates with respect to **I** are searched first, in order of decreasing size. If the largest subset template is reset, then all subset templates are reset. If all subset templates have been reset and if no other learned templates exist, then the first uncommitted node to be activated will code **I**. If all subset templates are searched and if there exist learned superset templates but no mixed templates, then the node with the smallest superset template will be activated next and will code **I**. If all subset templates are searched and if both superset templates $\mathbf{z}_J$ and mixed templates $\mathbf{z}_j$ exist, then u_j will be searched before u_J if and only if

$$|\mathbf{z}_j| < |\mathbf{z}_J| \qquad and \qquad \frac{|\mathbf{I}| - |\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{z}_J| - |\mathbf{z}_j|} < \frac{L_B}{L_A} \quad . \tag{1.71}$$

If all subset templates are searched and if there exist mixed templates but no superset templates, then a node u_j with a mixed template will be searched before an uncommitted node u_J if and only if

$$L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M > T_J(\mathbf{I}, t=0) \quad . \tag{1.72}$$

Where $T_J(\mathbf{I}, t=0) = L_A \sum I_i z_{iJ}(0) - L_B \sum z_{iJ}(0) + L_M$. The proof has several parts:

a) First we show that a node u_J with a subset template ($\mathbf{I} \cap \mathbf{z}_J = \mathbf{z}_J$) is searched before any node u_j with a non subset template. In this case,

$$O_j = L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M = |\mathbf{I} \cap \mathbf{z}_j|\left(L_A - L_B \frac{|\mathbf{z}_j|}{|\mathbf{I} \cap \mathbf{z}_j|}\right) + L_M \tag{1.73}$$

Now, note that

$$\frac{|\mathbf{z}_j|}{|\mathbf{I} \cap \mathbf{z}_j|} > \frac{N}{N-1} \tag{1.74}$$

because[8]

$$\left.\frac{|\mathbf{z}_j|}{|\mathbf{I} \cap \mathbf{z}_j|}\right|_{min} = \left.\frac{|\mathbf{z}_j|}{|\mathbf{z}_j| - 1}\right|_{min} = \frac{N-1}{N-2} > \frac{N}{N-1} \tag{1.75}$$

From eqs. (1.26), (1.69) and (1.74), it follows that

$$O_j < |\mathbf{I} \cap \mathbf{z}_j| L_B \left(\frac{L_A}{L_B} - \frac{N}{N-1}\right) + L_M < L_M \quad . \tag{1.76}$$

On the other hand,

$$O_J = (L_A - L_B)|\mathbf{z}_J| + L_M > L_M \tag{1.77}$$

Therefore,

$$O_J > O_j \tag{1.78}$$

b) Subset templates are searched in order of decreasing size:

Suppose two subset templates of **I**, $\mathbf{z}_J$ and $\mathbf{z}_j$ such that $|\mathbf{z}_J| > |\mathbf{z}_j|$. Then

8. We are assuming that u_j is not an uncommitted node ($|\mathbf{z}_j| < N$).

$$O_J = (L_A - L_B)|\mathbf{z}_J| + L_M > (L_A - L_B)|\mathbf{z}_j| + L_M = O_j \tag{1.79}$$

Therefore node u_J will be searched before node u_j. By eq. (1.61), if the largest subset template is reset, then all other subset templates are reset as well.

c) Subset templates u_J are searched before an uncommitted node u_j:

$$\begin{aligned} O_j &= L_A|\mathbf{I}| - L_B N + L_M \le L_A(N-1) - L_B N + L_M = L_B\left(\frac{L_A}{L_B}(N-1) - N\right) + L_M < \\ &< L_B\left(\frac{N}{N-1}(N-1) - N\right) + L_M = L_M < (L_A - L_B)|\mathbf{z}_J| + L_M = O_J \end{aligned} \tag{1.80}$$

Therefore now, if all subset templates are searched and if no other learned template exists, then an uncommitted node will be activated and code the input pattern.

d) If all subset templates have been searched and there exist learned superset templates but no mixed templates, the node with the smallest superset template u_J will be activated (and not an uncommitted node u_j) and code $\mathbf{I}$:

$$O_J = L_A|\mathbf{I}| - L_B|\mathbf{z}_J| + L_M > L_A|\mathbf{I}| - L_B N + L_M = O_j \tag{1.81}$$

If there are more than one superset templates, the one with the smallest $|\mathbf{z}_J|$ will be activated. Since $|\mathbf{I} \cap \mathbf{z}_J| = |\mathbf{I}| \ge \rho|\mathbf{I}|$ there is no reset, and $\mathbf{I}$ will be coded.

e) If all subset templates have been searched and there exist a superset template u_J and a mixed template u_j, then $O_j > O_J$ if and only if eq. (1.71) holds:

$$O_j - O_J = L_A(|\mathbf{I} \cap \mathbf{z}_j| - |\mathbf{I}|) + L_B(|\mathbf{z}_J| - |\mathbf{z}_j|) \tag{1.82}$$

e.1) if eq. (1.71) holds:

$$O_j - O_J = L_A\left(\frac{L_B}{L_A} - \frac{|\mathbf{I}| - |\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{z}_J| - |\mathbf{z}_j|}\right)(|\mathbf{z}_J| - |\mathbf{z}_j|) > 0 \tag{1.83}$$

e.2) if $O_j > O_J$:

Assume first that $|\mathbf{z}_J| - |\mathbf{z}_j| < 0$. Then, by eq. (1.83), it has to be

$$\frac{L_B}{L_A} < \frac{|\mathbf{I}| - |\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{z}_J| - |\mathbf{z}_j|} \tag{1.84}$$

Since $L_A > L_B > 0$ it had to be $|\mathbf{I}| - |\mathbf{I} \cap \mathbf{z}_j| < 0$, which is false. Therefore, it must be $|\mathbf{z}_J| - |\mathbf{z}_j| > 0$ and

$$\frac{L_B}{L_A} > \frac{|\mathbf{I}| - |\mathbf{I} \cap \mathbf{z}_j|}{|\mathbf{z}_J| - |\mathbf{z}_j|} \tag{1.85}$$

f) If all subset templates are searched and if there exist mixed templates but no superset templates, then a node u_j with a mixed template ($O_j = L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M$) will be searched before an uncommitted node u_J ($O_J = L_A|\mathbf{I}| - L_B N + L_M$) if and only if eq. (1.72) holds:

$$\begin{aligned} O_j - O_J &= L_A\left(|\mathbf{I} \cap \mathbf{z}_j| - |\mathbf{I}|\right) - L_B\left(|\mathbf{z}_j| - N\right) > 0 \Leftrightarrow \\ \Leftrightarrow L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M &> L_A|\mathbf{I}| - L_B N + L_M = T_J(\mathbf{I}, \text{t=0}) \end{aligned} \tag{1.86}$$

This completes the proof of the modified Theorem 7 for the modified ART 1 architecture.

I. Biasing the Network towards Uncommitted Nodes:

In the original ART 1 architecture, choosing L large increases the tendency of the network to choose uncommitted nodes in response to unfamiliar input patterns $\mathbf{I}$. In the modified ART 1 architecture, the same effect is observed when choosing L_A/L_B large. This can be understood through the following reasoning.

When an input pattern $\mathbf{I}$ is presented, an uncommitted node is chosen before a coded node u_j if

$$L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| < L_A|\mathbf{I}| - L_B N \tag{1.87}$$

This inequality is equivalent to

$$\frac{L_A}{L_B} > \frac{N - |\mathbf{z}_j|}{|\mathbf{I}| - |\mathbf{I} \cap \mathbf{z}_j|} \tag{1.88}$$

As the ratio L_A/L_B increases it is more likely that eq. (1.88) is satisfied, and hence that uncommitted nodes are chosen before coded nodes, regardless of the *vigilance parameter* value ρ.

J. Remarks:

Even though in this Section we have showed that the computational properties of the original ART 1 system are preserved in the modified ART 1 system, the response of both systems to an arbitrary list of training patterns will not be exactly the same. The main underlying reason for this difference in behavior is that the initial ordering

$$O_{j_1} > O_{j_2} > O_{j_3} > \ldots \tag{1.89}$$

is not always exactly the same for both architectures. In the next Section we will try to study the differences between the two ART 1 systems. As we will see, for most cases the behavior is identical, although in a few cases a different behavior results.

1.6. Functional Differences between Original and Modified Model

As stated previously, the difference in behavior between the ART 1 and ART 1_m models is caused by the different orderings of the terms of eq. (1.89). Assuming that both models, at a certain time, have identical weight templates $\{\mathbf{z}_j\}$, and the same input pattern $\mathbf{I}$ is given, eq. (1.89) has the following two formulations:

$$\text{Original ART 1:} \quad \frac{|\mathbf{I} \cap \mathbf{z}_{j_1}|}{L-1+|\mathbf{z}_{j_1}|} > \frac{|\mathbf{I} \cap \mathbf{z}_{j_2}|}{L-1+|\mathbf{z}_{j_2}|} > \frac{|\mathbf{I} \cap \mathbf{z}_{j_3}|}{L-1+|\mathbf{z}_{j_3}|} > \ldots$$
$$\text{Modified ART 1:} \quad \frac{L_A}{L_B}|\mathbf{I} \cap \mathbf{z}_{l_1}| - |\mathbf{z}_{l_1}| > \frac{L_A}{L_B}|\mathbf{I} \cap \mathbf{z}_{l_2}| - |\mathbf{z}_{l_2}| > \ldots \tag{1.90}$$

where j_k might be different than l_k. The ordering resulting for the original ART 1 description is modulated by parameter $L > 1$. For example, if L is very large compared to all $|\mathbf{z}_j|$ terms, then the ordering depends exclusively on the values of $|\mathbf{I} \cap \mathbf{z}_j|$,

$$|\mathbf{I} \cap \mathbf{z}_{j_1}| > |\mathbf{I} \cap \mathbf{z}_{j_2}| > |\mathbf{I} \cap \mathbf{z}_{j_3}| > \ldots \tag{1.91}$$

If L is very close to 1, then the ordering depends on the ratios,

$$\frac{|\mathbf{I} \cap \mathbf{z}_{j_1}|}{|\mathbf{z}_{j_1}|} > \frac{|\mathbf{I} \cap \mathbf{z}_{j_2}|}{|\mathbf{z}_{j_2}|} > \frac{|\mathbf{I} \cap \mathbf{z}_{j_3}|}{|\mathbf{z}_{j_3}|} > \ldots \tag{1.92}$$

Likewise, for the ART 1_m description, the ordering is modulated by a single parameter $\alpha = L_A/L_B > 1$. If α is extremely large, the situation in eq. (1.91) results. However, for α very close to 1, the ordering depends on the differences,

$$|\mathbf{I} \cap \mathbf{z}_{l_1}| - |\mathbf{z}_{l_1}| > |\mathbf{I} \cap \mathbf{z}_{l_2}| - |\mathbf{z}_{l_2}| > |\mathbf{I} \cap \mathbf{z}_{l_3}| - |\mathbf{z}_{l_3}| > \ldots \tag{1.93}$$

Obviously, the behavior of the two ART 1 descriptions will be identical for large values of L and α. However, moderate values of L and α are desired in practical ART 1 applications. On the other hand, it can be expected that the behavior will also tend to be similar for very high values of ρ: if ρ is very close to 1, each training pattern will form an independent category. However, different training patterns will cluster into a shared category for smaller values of ρ. Therefore, a very similar behavior between ART 1 and ART 1_m will be expected for high values of ρ, while more differences in behavior might be apparent for smaller values of ρ.

In order to compare the two algorithms' behavior, we have performed exhaustive simulations using randomly generated training patterns sets[9]. As an illustration of a typical case where the two algorithms produce different learned templates, Fig. 1.5 shows the evolution of the memory templates, for both the ART 1 and the ART 1_m algorithms, using a randomly generated training set of 10 patterns with 25 pixels each. Weight templates for original ART 1 are named $\mathbf{z}_j$, while for ART 1_m they are named $\mathbf{z}_j'$. The vigilance parameter was set to $\rho = 0.4$ for the original ART 1 $L = 5$, and for the ART 1_m $\alpha = 2$. In Fig. 1.5, boxed category templates are those that met the vigilance criterion and had the maximum T_j value. If the box is drawn with a continuous line, the corresponding $\mathbf{z}_j$ template suffered modifications due to learning. If the box is drawn with dashed line, learning did not alter the corresponding $\mathbf{z}_j$ template. Both algorithms stabilized their weights in 2 training trials. Looking at the learned templates we can see that input patterns 4 and 5 clustered in the same category for both algorithms ($\mathbf{z}_4$ for original ART 1 and $\mathbf{z}'_3$ for ART 1_m). This also occurred for patterns 6 and 8 ($\mathbf{z}_3$ and $\mathbf{z}'_2$) and for patterns 3, 9 and 10 ($\mathbf{z}_5$ and $\mathbf{z}'_5$). However, patterns 1, 2,

9. For all simulations in this Appendix, randomly generated training patterns sets were obtained with a 50% probability for a pixel to be either '1' or '0'.

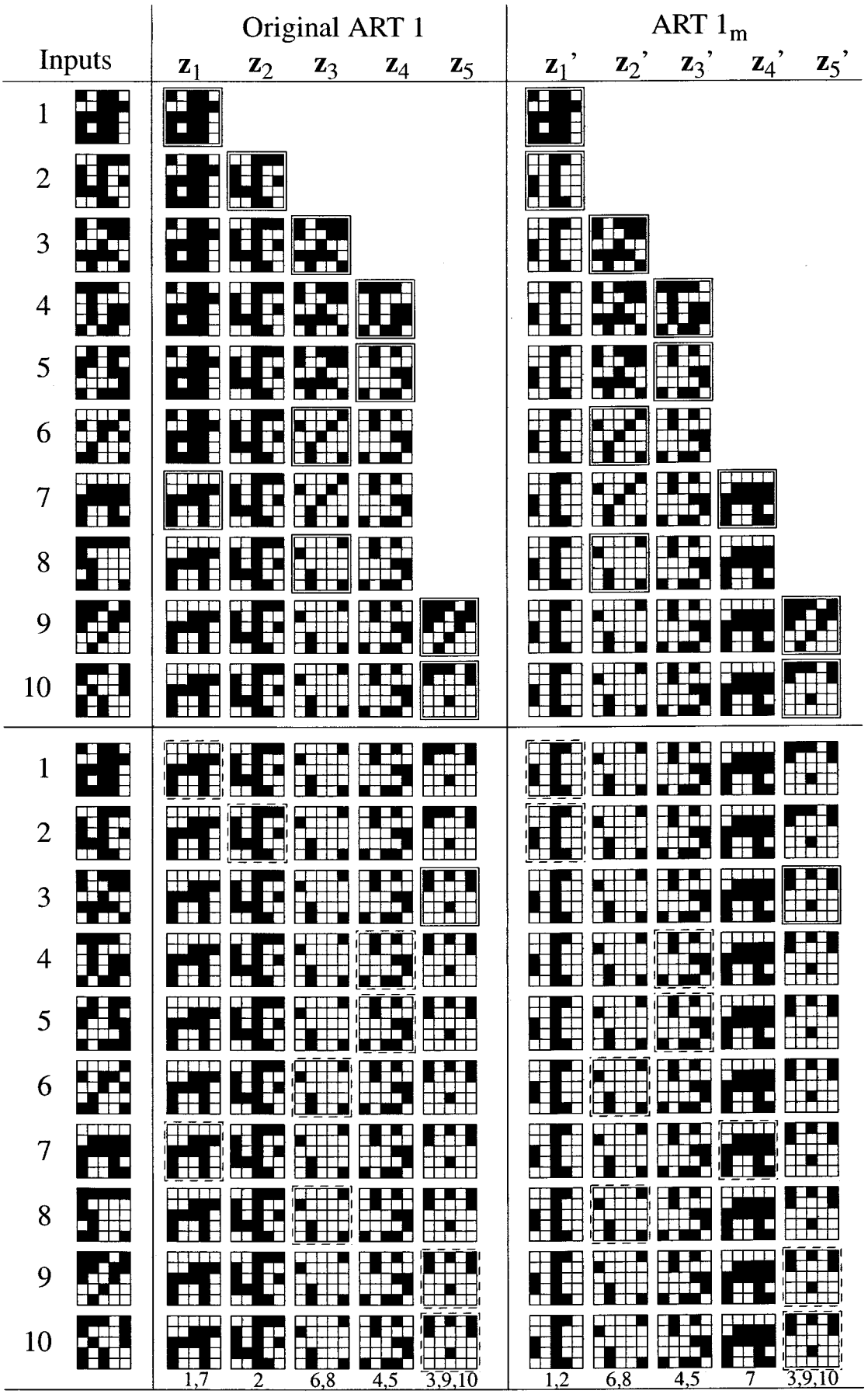


Fig. 1.5: Comparative Learning Example (ρ=0.4, L=5, α=2)

and 7 did not cluster in the same way in the two cases. In the original ART 1 algorithm patterns 1 and 7 clustered into category $\mathbf{z}_1$, while pattern 2 remained independent in category $\mathbf{z}_2$. In the ART 1_m algorithm patterns 1 and 2 clustered together into category $\mathbf{z'}_1$, while pattern 7 remained independent in category $\mathbf{z'}_4$.

To measure a distance between the two templates $\mathbf{z}_j$ and $\mathbf{z'}_j$, let us use the Hamming distance between two binary patterns $\mathbf{a} \equiv (a_1, a_2, \ldots a_N)$ and $\mathbf{b} \equiv (b_1, b_2, \ldots b_N)$,

$$d(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{N} f_d(a_i, b_i) , \tag{1.94}$$

where

$$f_d(a_i, b_i) = \begin{cases} 0 & \text{if} \quad a_i = b_i , \\ 1 & \text{if} \quad a_i \neq b_i . \end{cases} \tag{1.95}$$

We can use this metric to define the distance between two sets of patterns $\{\mathbf{z}_j\}_{j=1}^{Q}$ and $\{\mathbf{z'}_j\}_{j=1}^{Q}$ as that which minimizes

$$\sum_{i=1}^{Q} d(\mathbf{z}_i, \mathbf{z}_{l_i}') . \tag{1.96}$$

For this purpose, the optimal ordering of indexes $(l_1, l_2, \ldots l_Q)$ must be found. In the case of Fig. 1.5 (where $Q = 5$), the distance D between the two learned patterns sets is given by,

$$D = d(\mathbf{z}_1, \mathbf{z'}_4) + d(\mathbf{z}_2, \mathbf{z'}_1) + d(\mathbf{z}_3, \mathbf{z'}_2) + d(\mathbf{z}_4, \mathbf{z'}_3) + d(\mathbf{z}_5, \mathbf{z'}_5) = 7 . \tag{1.97}$$

In general, we can define the distance between two patterns sets $\mathbf{A} = \{\mathbf{a}_j\}_{j=1}^{Q}$ and $\mathbf{B} = \{\mathbf{b}_j\}_{j=1}^{Q}$ as,

$$D(\mathbf{A}, \mathbf{B}) = \min_{\{l_1, l_2, \ldots l_Q\}} \left[\sum_{i=1}^{Q} d(\mathbf{a}_i, \mathbf{b}_{l_i}) \right] . \tag{1.98}$$

In the case of Fig. 1.5, both algorithms produced the same number of learned categories. This does not always occur. For the case where a different number of categories results, we measured the distance between the two learned sets by adding as many uncommitted F_2 nodes to the set with less categories as necessary to equal the number of categories. An uncommitted category has all its pixels set to '1'. Thus, having a different number of committed nodes drastically increases the resulting distance, and is consequently a strong penalty.

We have repeated the simulation of Fig. 1.5 many times for different sets of randomly generated training patterns and sweeping the values of ρ, L, and α. For each combination of ρ, L, and α values, we repeated the simulation 100 times for different training patterns sets, and computed the average number of learned categories, learning trials, and distance between learned categories, as well as their corresponding standard deviations. Fig. 1.6 and Fig. 1.7 present the results of these simulations. Fig. 1.6(a) shows how the average number of learned categories changes with L (from 1.01 to 40) for different values of ρ, for the original ART

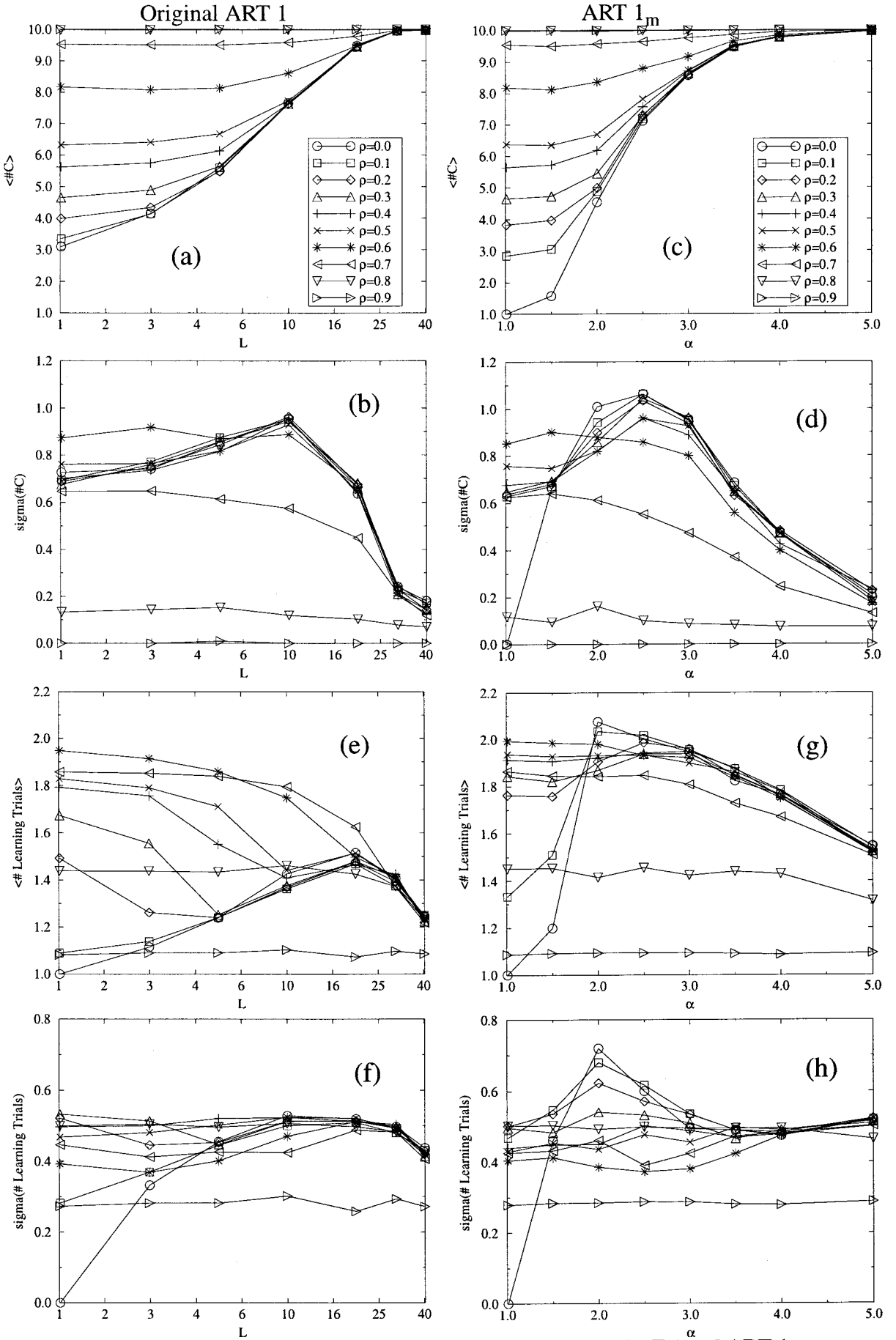


Fig. 1.6: Simulated Results Comparing Behavior between ART 1 and ART 1_m

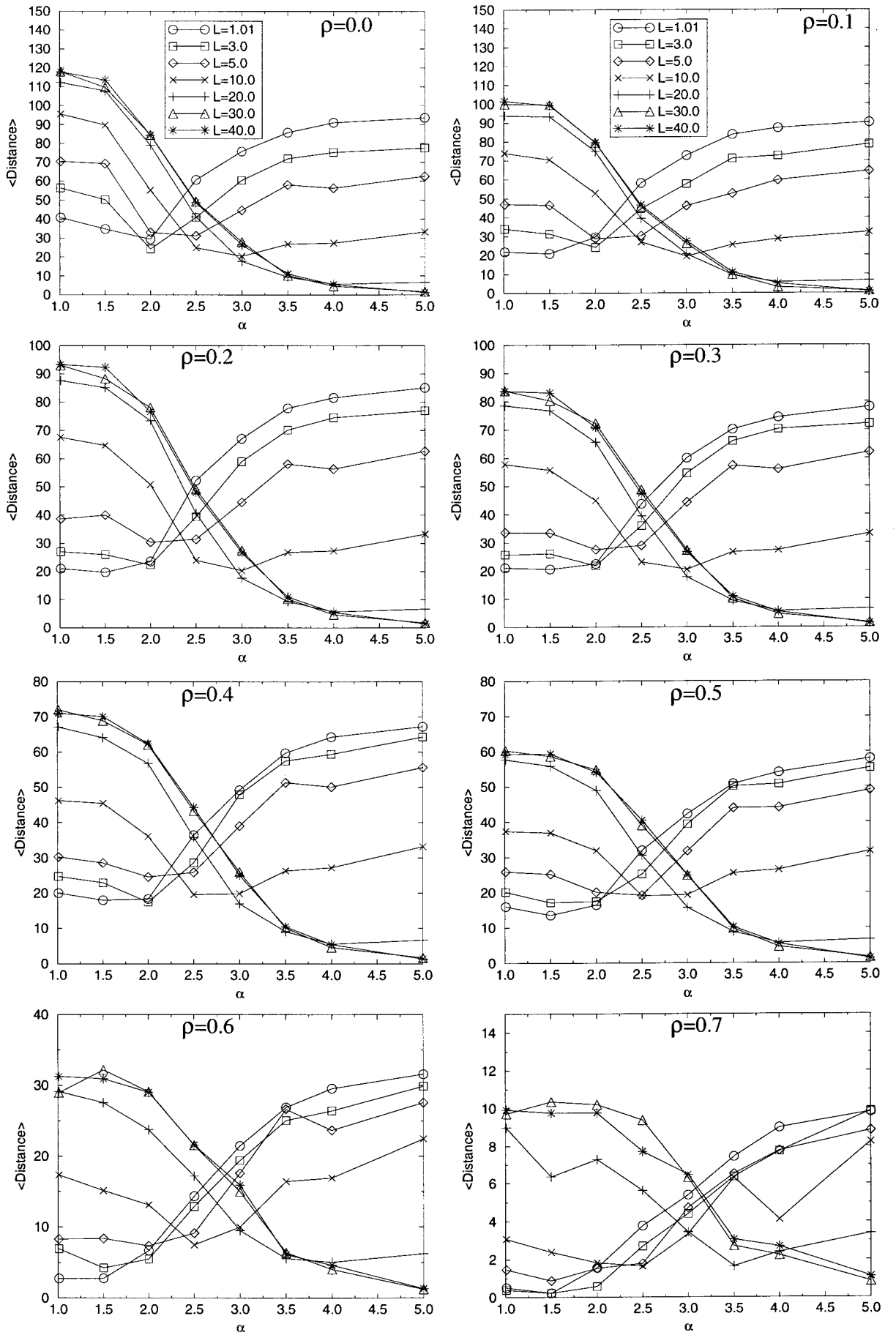


Fig. 1.7: Learned Categories Average Distances

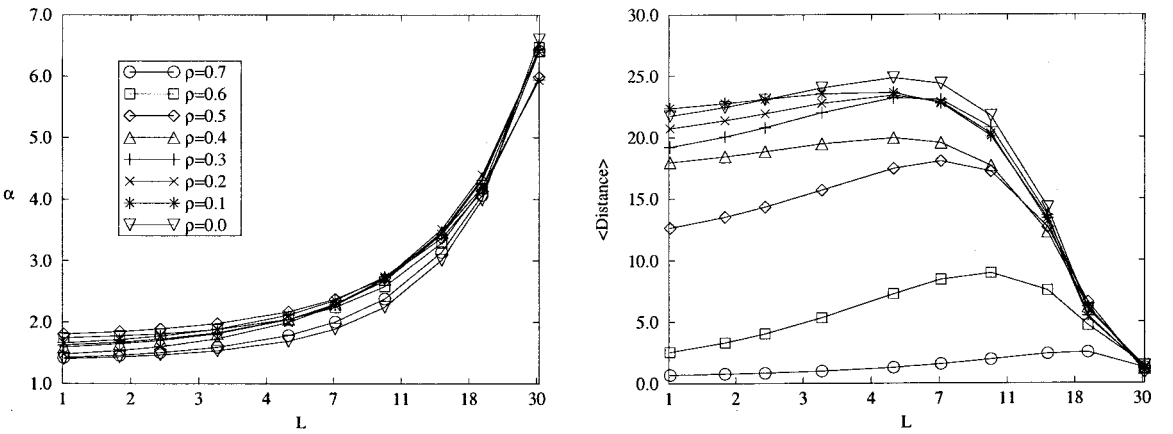


Fig. 1.8: Optimal parameters fit between ART 1 and ART 1_m

1. As ρ decreases, parameter L has more control on the average number of learned categories. Fig. 1.6(b) shows the standard deviation for the number of learned categories of Fig. 1.6(a). As the number of learned categories approaches the number of training patterns (10 in this case), standard deviation decreases. This happens for large values of L (independently of ρ) and for large values of ρ (independently of L). Fig. 1.6(c) and Fig. 1.6(d) show the same as Fig. 1.6(a) and Fig. 1.6(b) respectively, for the ART 1_m algorithm. As we can see, parameter α (swept from 1.01 to 5.0) of ART 1_m has more tuning power than parameter L of the original ART 1. On the other hand, ART 1_m presents a slightly higher standard deviation than the original ART 1. Nevertheless, the qualitative behavior of both algorithms is similar. Fig. 1.6(e) and Fig. 1.6(f) show the average number of learning trials and their corresponding deviations, needed by the original ART 1 algorithm to stabilize its learned weights. Fig. 1.6(g) and Fig. 1.6(h) show the same for the ART 1_m algorithm. As we can see, the ART 1_m algorithm needs a slightly higher average number of learning trials to stabilize. Also, the standard deviation observed for the ART 1_m algorithm is slightly higher. Finally, Fig. 1.7 shows the resulting average distances (as defined by eq. (1.98)) between learned categories of the ART 1 and the ART 1_m algorithms. For ρ changing from 0.0 to 0.7 in steps of 0.1, each sub-figure in Fig. 1.7 depicts the resulting average distance for different values of L while sweeping α between 1.01 and 5.0.

It seems natural to expect that, for a given value of ρ and a given value of the original ART 1 parameter L, there is an optimal value for the ART 1_m parameter α that will minimize the difference in behavior between the two algorithms. To find this relation between L and α for each ρ, we computed (for a given ρ and L) the value of α that minimizes the average distance between the learned patterns sets generated by the two algorithms. The results of these computations are shown in Fig. 1.8 [10]. Fig. 1.8(a) shows a family of curves (one for each value of ρ), that shows the optimal value of α as a function of L. Fig. 1.8(b) shows the resulting minimum average distance between learned sets for the same family of curves. As shown in Fig. 1.8(a), the optimum fit between parameters α and L is very slightly dependent on the value of ρ.

As can be concluded from Fig. 1.6, Fig. 1.7, Fig. 1.8, and the discussion in this Section, the behavior of the two algorithms is qualitatively the same although some slight quantitative differences can be observed. ART 1_m parameter α has a wider tuning range than original ART 1 parameter L. On the other hand, ART 1_m needs a slightly higher number of learning trials than the original ART 1. Also, there is an optimal adjustment

10. Note that high values of ρ and L were omitted in this analysis, since in these cases the behavior of the two algorithms tends to be similar, regardless of the fit between parameters L and α.

between parameters α and L that minimizes the difference in behavior between the two algorithms, and this adjustment appears approximately independent of ρ.

1.7. Extending the ART 1_m Model to *Type-2* and *Type-1* Descriptions

The great advantage of the ART 1_m algorithm is its ability to produce a very simple *Type-3* hardware implementation, requiring only a binary valued memory template and only addition, subtraction and comparison operations, as well as a Winner-Take-All competition. Although *Type-2* and *Type-1* descriptions can be found that lead to the *Type-3* behavior of the ART 1_m algorithm described in this paper, these descriptions do not possess the hardware-attractive features of the *Type-3* implementation. Nevertheless, brief *Type-2* and a *Type-1* descriptions for this ART 1_m algorithm are presented in this Section.

A. A Type-2 ART 1_m Implementation

The change in weights must be smooth in a *Type-2* description. Every time an input pattern **I** is presented and an F_2 category node is selected for LTM storage, only a partial change in LTM traces is allowed. In this case, it is obvious that we can no longer use a binary valued weight template.

As seen in Section 1.4, Fig. 1.3(c) shows the flow diagram of a *Type-3* implementation of the ART 1_m algorithm. Extending this diagram to a *Type-2* description is straightforward. The only box that needs to be changed is that corresponding to the update of weights. Instead of using the algebraic formula $\mathbf{z}_J(new) = \mathbf{I} \cap \mathbf{z}_J(old)$ we have to use a time domain differential equation that would lead to the same steady state. The following set of differential equations fulfills this requirement,

$$\dot{z}_{ij} = Kf(y_j)\,[-z_{ij} + h(x_i)]\,, \tag{1.99}$$

where K is a positive constant, $h(\cdot)$ a sigmoidal function, and x_i is the STM activity of node v_i in the F_1 layer, given by,

$$x_i = I_i \sum_j y_j z_{ij} = I_i z_{iJ}\,. \tag{1.100}$$

Note that eq. (1.100) has the same form than eq. (1.16) which describes the evolution of the top-down weights of the *Type-2* implementation of the original algorithm. Similarly eq. (1.100) is the result of substituting in eq. (1.13) the top-down weights of the original algorithm by the set of weights $\{z_{ij}\}$ of the modified algorithm.

If T_∞ is the time required for the LTM eqs. (1.99) to settle to their steady state, the update of weights (i.e., the simulation of eqs. (1.99)) would be allowed only for a time interval $\tau \ll T_\infty$ for each input pattern **I** presentation. As τ approaches T_∞, application of eqs. (1.99) or the update weights equation of Fig. 1.3(c) would become equivalent. Fig. 1.9 shows the flow diagram corresponding to this *Type-2* implementation of the ART 1_m algorithm.

B. A Type-1 ART 1_m Implementation

For a *Type-1* implementation, an appropriate set of STM equations must be found that leads to the flow diagram of Fig. 1.9 when the STM time constants are very small compared to the LTM ones. The following time domain STM differential equations would serve our purpose,

$$\begin{aligned} F_1: \quad & \varepsilon \dot{x}_i = -x_i + (1 - A_1 x_i) J_i^+ - (B_1 + C_1 x_i) J_i^- \\ F_2: \quad & \varepsilon \dot{y}_j = -y_j + (1 - A_2 y_j) J_j^+ - (B_2 + C_2 y_j) J_j^- \end{aligned} \tag{1.101}$$

where,

$$\begin{aligned} J_i^+ &= I_i + D_1 \sum_j f(y_j) z_{ij} \quad , \\ J_i^- &= \sum_j f(y_j) \quad , \\ J_j^+ &= g(y_j) + T_j \quad , \\ J_j^- &= \sum_{k \neq j} g(y_k) \quad . \end{aligned} \tag{1.102}$$

Parameters ε, A_1, B_1, C_1, A_2, B_2, C_2, and D_1 are positive and constant. Functions $f(\cdot)$ and $g(\cdot)$ are sigmoidal. Functions $g(\cdot)$ will be responsible for the resulting Winner-Take-All action of the F_2 layer. These STM equations are identical to those of the original ART 1 algorithm [1.2], except that we use one weight

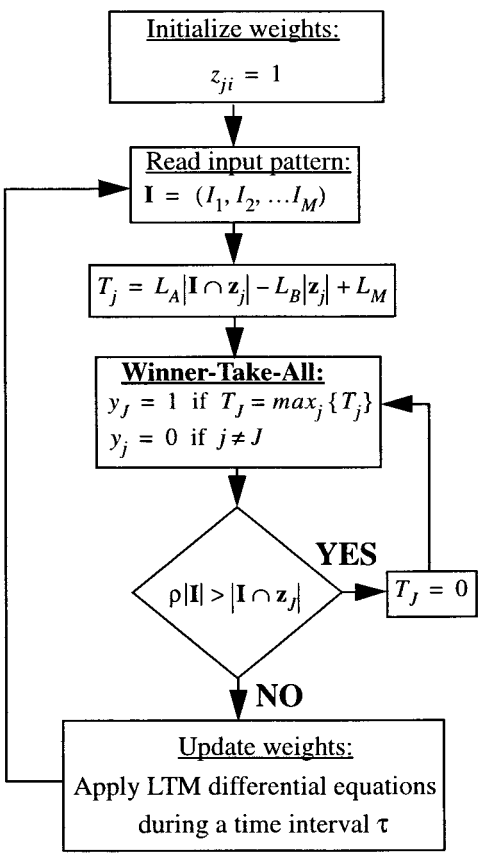


Fig. 1.9: ART 1_m algorithm *Type-2* implementation

template instead of two. However, the main difference lies in the way the terms T_j are computed. In this case T_j will be given by the following equation,

$$T_j = D_2\left[L_A \sum_i h(x_i)\, z_{ij} - L_B \sum_i z_{ij} + L_M\right]. \tag{1.103}$$

where D_2 is constant and positive. Using eqs. (1.101)-(1.103) together with an STM *Reset System* will assure that if the STM time constants are very small compared to the LTM ones, the *Type-2* description of Fig. 1.9 results. The *Reset System* can be identical to that used in the original ART 1 system: each active input ($I_i = 1$) sends an excitatory signal of size P to an orienting subsystem A. Each F_1 node x_i which exceeds zero generates an inhibitory signal of size Q and sends it to A. The orienting subsystem A generates a nonspecific reset wave to F_2 whenever

$$\frac{|\mathbf{X}|}{|\mathbf{I}|} < \rho = \frac{P}{Q}, \tag{1.104}$$

where $\mathbf{I}$ is the input pattern and $|\mathbf{X}|$ is the number of F_1 nodes such that $x_i > 0$. The nonspecific reset wave shuts off active F_2 nodes until the input pattern $\mathbf{I}$ shuts off.

1.8. Alternative ART 1 Modifications

Other alternatives to the computation of the terms T_j of eq. (1.25) have been proposed [1.8] for a Fuzzy-ART architecture. Since ART 1 reduces to a particular case of Fuzzy-ART when the input pattern $\mathbf{I}$ is binary valued, any valid way of computing T_j in Fuzzy-ART should, in principle, be valid for ART 1 as well. The different T_j functions (also called '*distances*' or '*choice functions*') proposed in [1.8] when particularized for ART 1 result in the following formulations:

$$\begin{aligned} &\text{Function 1:} \quad |\mathbf{I} \cap \mathbf{z}_j| - |\mathbf{z}_j| + \varepsilon\,(|\mathbf{z}_j| - |\mathbf{I} \cup \mathbf{z}_j|), \\ &\text{Function 2:} \quad |\mathbf{I} \cap \mathbf{z}_j| - |\mathbf{z}_j| + \varepsilon\,(|\mathbf{z}_j| - |\mathbf{I}|). \end{aligned} \tag{1.105}$$

Note that these functions are also based on the subtraction operation, as in ART 1_m, but are computationally more expensive since either $|\mathbf{I} \cup \mathbf{z}_j|$ or $|\mathbf{I}|$ has to be computed as well. The *choice function* that we have used in this paper would be equivalent to the following,

$$T_j = |\mathbf{I} \cap \mathbf{z}_j| - |\mathbf{z}_j| + \varepsilon|\mathbf{z}_j| = |\mathbf{I} \cap \mathbf{z}_j| - (1-\varepsilon)\,|\mathbf{z}_j|, \tag{1.106}$$

and parameter $\alpha = L_A/L_B > 1$ would have been equivalent to

$$\alpha = \frac{1}{1-\varepsilon}. \tag{1.107}$$

If all the original ART 1 properties are to be preserved, we know now that α has to be greater than one. This implies,

$$\alpha > 1 \quad \Leftrightarrow \quad 1 > \varepsilon > 0. \tag{1.108}$$

With respect to the *choice functions* in eq. (1.105), Function 2 is mathematically equivalent to eq. (1.106), because the only difference between the two is the term $-\varepsilon|\mathbf{I}|$. Since the input is common to all of the category nodes and does not change during a single presentation, this term effectively acts as a uniform negative bias on all of the category nodes, regardless of the pattern coded in their templates. Eq. (1.106), therefore, is more efficient because the input size computation is unnecessary.

Function 1 of eq. (1.105) is another valid *choice function*, but is also computationally more expensive than eq. (1.106). It can be shown that the original ART 1 computational properties are preserved when this function is used (provided $\varepsilon > 0$). To see this, substitute the equations of Section 1.5 whose numbers appear in the first column of Table 1 by the equations in the second column, and note that

$$\begin{aligned} |\mathbf{I} \cup \mathbf{z}_j| &\geq |\mathbf{z}_j|, |\mathbf{I}| \\ |\mathbf{I} \cap \mathbf{z}_j| &\leq |\mathbf{z}_j|, |\mathbf{I}| \\ |\mathbf{I} \cup \mathbf{z}_J| &= |\mathbf{I}| + |\mathbf{z}_J| - |\mathbf{I} \cap \mathbf{z}_J| \end{aligned} \tag{1.109}$$

are always satisfied (if we know that $\mathbf{I} \neq \mathbf{z}_j$ then the '$\geq$' and '$\leq$' signs in eq. (1.109) can be substituted by '$>$' and '$<$', respectively). Table 1.1 only provides the demonstrations for properties A, *B, E, G* and *I* of Section 1.5 Properties C, D and F are automatically satisfied since they do not depend on the explicit formulation of T_j. With respect to property *H* (Search Order) it can be shown that all of them are fulfilled if eqs. (1.69), (1.71), and (1.72) are changed to

$$\frac{1}{1-\varepsilon} < \frac{N}{N-1}, \tag{1.110}$$

$$|\mathbf{z}_j| < |\mathbf{z}_J| \qquad \text{and} \qquad \frac{|\mathbf{I} \cup \mathbf{z}_j| - |\mathbf{z}_j|}{|\mathbf{z}_J| - |\mathbf{z}_j|} < \varepsilon, \text{ and} \tag{1.111}$$

$$|\mathbf{I} \cap \mathbf{z}_j| - \varepsilon|\mathbf{I} \cup \mathbf{z}_j| - (1-\varepsilon)|\mathbf{z}_j| > T_J(\mathbf{I}, \text{t=0}), \tag{1.112}$$

respectively.

1.9. References

[1.1] T. Serrano-Gotarredona and B. Linares-Barranco, "A Modified ART 1 Algorith more suitable for VLSI Implementations," *Neural Networks*, 1996

[1.2] G. A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer Vision, Graphics, and Image Processing*, vol. 37, pp. 54-115, 1987.

[1.3] G. A. Carpenter and S. Grossberg, "ART 2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns," *Applied Optics*, vol. 26, No. 23, pp. 4919-4930, 1 December 1987.

[1.4] G. A. Carpenter and S. Grossberg, "ART 3: Hierarchical Search Using Chemical Transmitters in Self-Organizing Pattern Recognition Architectures," *Neural Networks*, vol. 3, pp. 129-152, 1990.

[1.5] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System," *Neural Networks*, vol. 4, pp. 759-771, 1991.

[1.6] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network," *Neural Networks*, vol. 4, pp. 565-588, 1991.

[1.7] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. B. Rosen, "Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps," *IEEE Transactions on Neural Networks*, vol. 3, No. 5, pp. 698-712, September 1992.

[1.8] G. A. Carpenter and M. N. Gjaja, "Fuzzy ART Choice Functions," *Proceedings of the 1994 World Congress on*

original equation	new equation
(1.35)	$T_{j_1} = \lvert\mathbf{z}_{j_1}\rvert - \varepsilon\lvert\mathbf{z}_{j_1}\rvert - (1-\varepsilon)\lvert\mathbf{z}_{j_1}\rvert = 0$ $T_{j_2} = \lvert\mathbf{z}_{j_1}\rvert - \varepsilon\lvert\mathbf{z}_{j_2}\rvert - (1-\varepsilon)\lvert\mathbf{z}_{j_2}\rvert = \lvert\mathbf{z}_{j_1}\rvert - \lvert\mathbf{z}_{j_2}\rvert < 0$
(1.36)	$T_{j_1} = \lvert\mathbf{z}_{j_1}\rvert - \varepsilon\lvert\mathbf{z}_{j_2}\rvert - (1-\varepsilon)\lvert\mathbf{z}_{j_1}\rvert = \varepsilon(\lvert\mathbf{z}_{j_1}\rvert - \lvert\mathbf{z}_{j_2}\rvert) < 0 \quad \text{if} \quad \varepsilon > 0$ $T_{j_2} = \lvert\mathbf{z}_{j_2}\rvert - \varepsilon\lvert\mathbf{z}_{j_2}\rvert - (1-\varepsilon)\lvert\mathbf{z}_{j_2}\rvert = 0$
(1.41)(1.42)	$T_J = \lvert\mathbf{I}\rvert - \varepsilon\lvert\mathbf{I}\rvert - (1-\varepsilon)\lvert\mathbf{I}\rvert = 0$ $T_j = \lvert\mathbf{I}\cap\mathbf{z}_j\rvert - \lvert\mathbf{z}_j\rvert + \varepsilon(\lvert\mathbf{z}_j\rvert - \lvert\mathbf{I}\cup\mathbf{z}_j\rvert) < 0 \quad \text{if} \quad \varepsilon > 0$
(1.49)	$T_J(new) = \lvert\mathbf{I}\cap\mathbf{z}_J(new)\rvert - \varepsilon\lvert\mathbf{I}\cup\mathbf{z}_J(new)\rvert - (1-\varepsilon)\lvert\mathbf{z}_J(new)\rvert =$ $= \lvert\mathbf{I}\cap\mathbf{z}_J(old)\rvert - \varepsilon\lvert\mathbf{I}\cup[\mathbf{I}\cap\mathbf{z}_J(old)]\rvert - (1-\varepsilon)\lvert\mathbf{I}\cap\mathbf{z}_J(old)\rvert =$ $= \lvert\mathbf{I}\cap\mathbf{z}_J(old)\rvert - \varepsilon\lvert\mathbf{I}\rvert - (1-\varepsilon)[\lvert\mathbf{I}\rvert + \lvert\mathbf{z}_J(old)\rvert - \lvert\mathbf{I}\cup\mathbf{z}_J(old)\rvert] \geq$ $\geq \lvert\mathbf{I}\cap\mathbf{z}_J(old)\rvert - \varepsilon\lvert\mathbf{I}\cup\mathbf{z}_J(old)\rvert - (1-\varepsilon)\lvert\mathbf{z}_J(old)\rvert = T_J(old)$
(1.59)	$O_j = \varepsilon\lvert\mathbf{z}_j\rvert - \varepsilon\lvert\mathbf{I}\rvert < O_J = \varepsilon\lvert\mathbf{z}_J\rvert - \varepsilon\lvert\mathbf{I}\rvert \quad \Leftrightarrow \quad \lvert\mathbf{z}_j\rvert < \lvert\mathbf{z}_J\rvert \quad (\varepsilon > 0)$
(1.62)	$O_j = \varepsilon\lvert\mathbf{z}_j\rvert - \varepsilon\lvert\mathbf{I}\rvert < O_J = \lvert\mathbf{I}\cap\mathbf{z}_J\rvert - \varepsilon\lvert\mathbf{I}\cup\mathbf{z}_J\rvert - (1-\varepsilon)\lvert\mathbf{z}_J\rvert <$ $< \lvert\mathbf{z}_J\rvert - \varepsilon\lvert\mathbf{I}\rvert - (1-\varepsilon)\lvert\mathbf{z}_J\rvert = \varepsilon\lvert\mathbf{z}_J\rvert - \varepsilon\lvert\mathbf{I}\rvert \quad \Rightarrow \quad \lvert\mathbf{z}_j\rvert < \lvert\mathbf{z}_J\rvert \quad (\varepsilon > 0)$
(1.63)	$O_j = \lvert\mathbf{z}_j\rvert - \varepsilon\lvert\mathbf{I}\rvert - (1-\varepsilon)\lvert\mathbf{z}_j\rvert < \lvert\mathbf{I}\cap\mathbf{z}_J\rvert - \varepsilon\lvert\mathbf{I}\rvert - (1-\varepsilon)\lvert\mathbf{z}_J\rvert <$ $< \rho\lvert\mathbf{I}\rvert - \varepsilon\lvert\mathbf{I}\rvert - (1-\varepsilon)\lvert\mathbf{z}_j\rvert \quad \Rightarrow \quad \lvert\mathbf{z}_j\rvert < \rho\lvert\mathbf{I}\rvert$
(1.87)	$\lvert\mathbf{I}\cap\mathbf{z}_j\rvert - \varepsilon\lvert\mathbf{I}\cup\mathbf{z}_j\rvert - (1-\varepsilon)\lvert\mathbf{z}_j\rvert < \lvert\mathbf{I}\rvert - N$
(1.88)	$\varepsilon > \dfrac{\lvert\mathbf{I}\cap\mathbf{z}_j\rvert + N - \lvert\mathbf{I}\rvert - \lvert\mathbf{z}_j\rvert}{\lvert\mathbf{I}\cup\mathbf{z}_j\rvert - \lvert\mathbf{z}_j\rvert}$

Table 1.1

Neural Networks (WCNN'94), vol. I, pp. 713-722.

[1.9] K. Bult and H. Wallinga, "A Class of Analog CMOS Circuits Based on the Square-Law Characteristic of an MOS Transistor in Saturation," *IEEE Journal of Solid-State Circuits*, vol. SC-22, No. 3, pp. 357-365, 1987.

[1.10] J. J. F. Cavanagh, *Digital Computer Arithmetic*, McGraw-Hill, 1985.

[1.11] B. Gilbert, "Current-Mode Circuits From a Translinear Viewpoint: A Tutorial," in *Analogue IC Design: The Current-Mode Approach*, C. Toumazou, F. J. Lidgey, and D. G. Haigh (Eds.), IEE Circuits and Systems Series 2, Peter Peregrinus Ltd., London, UK, 1990, Chapter 2, pp. 11-91.

[1.12] E. Sánchez-Sinencio, J. Ramírez-Angulo, B. Linares-Barranco, and A. Rodríguez-Vázquez, "Operational Transconductance Amplifier-Based Nonlinear Function Synthesis," *IEEE Journal of Solid-State Circuits*, vol. 24, No. 6, pp. 1576-1586, 1989.

[1.13] T. Serrano-Gotarredona, B. Linares-Barranco, and J. L. Huertas, "A CMOS VLSI Analog Current-Mode High-Speed ART 1 Chip," *Proceedings of the 1994 IEEE International Conference on Neural Networks (ICNN'94)*, Orlando, Florida, vol. 3, pp. 1912-1916.

[1.14] T. Serrano-Gotarredona and B. Linares-Barranco, "A Real-Time Clustering Microchip Neural Engine," *IEEE Transactions on VLSI Systems,* 1996.

[1.15] D. H. Sheingold, *Nonlinear Circuits Handbook,* Analog Devices Inc., Norwood, Massachusetts, U. S. A., 1976.

Appendix 2: A Real-Time Clustering Microchip Neural Engine

2.1. Hardware Oriented Attractive Properties of the ART 1 Algorithm

Two types of neural hardware engineers can be distinguished. The first designs "*general purpose*" hardware accelerators or systems that speed up neural algorithms running on conventional computers [2.5]-[2.13]. This kind of hardware allows considerable flexibility in the topology and operations of the neural systems. In this way algorithm researchers have a powerful tool to further develop neural algorithms and industry engineers have some attractive chips that significantly speed up their neural commercial products. The second type of hardware engineers are those who design a real-time system for a specific application. They must select the best-suited algorithm and map it into hardware. This achieves a close-to-optimum efficient hardware for a limited range of applications. The work described in this appendix falls into this second category of hardware engineering. The specific application is real-time clustering of binary input patterns.

A clustering device is a device able to build categories from a collection of patterns. A **real-time** clustering device has to be able to do this at the speed of arrival of the patterns. There are some clustering algorithms [2.14]-[2.19] that need to be trained off-line to build the categories. For a real-time clustering device, however, it would be desirable to use an algorithm that can be trained on-line: if a new pattern arrives the algorithm updates it internal knowledge (instead of erasing all the accumulated knowledge and retrain with the old and new collection of patterns).

For the second type of neural hardware engineers, the issue of efficiently implementing in hardware a real size neural network is not a trivial task. Many neural network algorithms are available in the literature which have been developed, studied, and optimized for applications through computer and/or software based systems. Consequently, when designing a hardware realization, engineers face many problems like excessive interconnectivity, high resolution of weights, high precision of operations, complicated operator requirements (e.g., integrals and derivatives), high number of neurons required for a real-world application, etc. Many times some of these requirements can be relaxed, the topology modified, or the operations simplified, with no significant deterioration of global operation of the neural system but with a considerable boost in the hardware performance. Modifying neural algorithms to make them more VLSI-friendly and produce more efficient hardware should be a common practice among neural hardware engineers of the second type [2.20]-[2.23]. After selecting an appropriate neural algorithm the next step consists of studying how far the algorithm can be simplified without performance degradation. The simplifications have to be hardware-oriented, so that the final combination of "*theoretical algorithm*" + "*hardware circuit technique*" results in a high performance real time system. The success of the hardware system depends on the selection of the algorithm, the selection of a powerful circuit design technique, and how the algorithm is modified to efficiently "*marry*" the circuit technique resulting in an optimum performance final system.

In the case of our application, real-time binary patterns clustering, we chose the ART 1 algorithm mainly due to the attractive hardware-oriented properties (which will be highlighted bellow), as well as the theoretical computational properties (see Appendix 1)[2.3]. We also chose to slightly modify the

---

This Appendix is an amplified version of the paper [2.1].

mathematical ART 1 algorithm to obtain more efficient hardware. This modification (described in the previous Appendix) allows the use of simpler operations while preserving all the computational properties of the original ART 1 architecture [2.3], [2.4]. As an extra bonus, the hardware circuit introduces a significant speed improvement as it automatically parallels the sequential ART search process [2.2] inherent in the mathematical neural algorithm.

In performance comparison of hardware implementations, a common figure of merit is the number of interconnections per second. More refined figures have been proposed that include resolution and precision [2.24]. However, these figures would be reasonably fair criteria for the first type of hardware engineering mentioned above, the *general-purpose* one. In order to compare hardware systems of the second type, the *specific-application* neural hardware, some global figure must be used that evaluates the overall system performance. Usually this figure will be application dependent. In our case, since we are concerned with a real-time clustering application of binary input patterns, an appropriate figure of merit might be

$$ppc/s = \frac{\text{number of } \mathbf{p}\text{atterns processed}}{\text{seconds}} \times \mathbf{p}\text{ixels} \times \mathbf{c}\text{ategories} \tag{2.1}$$

where,

- **number of patterns processed/second** is the speed at which patterns are classified and learned (including the number of learning trials required). This speed generally depends on the patterns themselves, and on the knowledge already stored in the system. Therefore, this speed can be given as an average or as the slowest case measured.
- **pixels** is the maximum number of pixels of the input patterns.
- **categories** is the maximum number of categories the system is able to form.

As we will see later in the Section on experimental results, the chip described here is able to cluster up to 18 different categories of binary patterns with 100 pixels, while classifying and learning each pattern in less than 1.8μs. Since ART 1 learns on-line, 1 iteration of input patterns presentations provides the system with sufficient knowledge to perform properly[1]. This results in a *ppc/s* of

$$ppc/s = \frac{n \text{ patterns}}{1 \text{ iteration} \times n \text{ patterns} \times 1.8\mu s} \times 100 \text{ pixels} \times 18 \text{ categories} = 1.0\times10^{9} ppc/s \tag{2.2}$$

If we would like to obtain the same performance using *Backpropagation* based hardware, and assuming the network would learn with 10,000 iterations of patterns presentations, this means that a speed of 180*ps* would be needed for each pattern classification and corresponding weights update. Assuming this task could be performed with a Backpropagation network with 100 input neurons, 5 hidden-layer neurons, and 5 output neurons[2] (which means a total of $100 \times 5 + 5 \times 5 = 525$ interconnections), and that the speed of feedforward classification is the same as for feedback learning, hardware able to perform

---

1. The input patterns set can be iterated several times to stabilize the internal weights, but this is not necessary for the system to start working.

2. Optimistically, a backpropagation net with 5 output nodes might be able to code up to 2^5 categories.

$$\frac{2 \times 525 \text{ connections}}{180ps} = 5.83\times10^{12} \text{connections/s plus connection-updates/s} \quad (2.3)$$

would be needed. For the chip described here, since it is based on the powerful ART 1 algorithm, the above performance can be achieved with a hardware of only 4.4×10^{9} *connections/s* plus *connections-updates/s*, as discussed in the Subsection B of Section 2.3.

Note that the Backpropagation algorithm is not appropriate for clustering applications, and comparing it against ART 1 is slightly unfair. There are other algorithms available in the literature that have been developed specially for clustering applications [2.14]-[2.19]. However, they usually do not provide all the computational properties mentioned in Appendix 1, specially the "*On-Line Learning*" property which is crucial for real-time clustering, or they present serious difficulties when mapped into hardware.

Another hardware attractive feature that an ART 1 based implementation offers with respect to others, is that the interconnection weights do not have to be analog, as shown in the previous Appendix. Most of the neural algorithms reported in the literature require a real-valued set of weights defined within a certain interval. These weights can be discretized in a number of digital steps, but the granularity required for proper operation of the system is usually very fine (around 16-bits for the Back-Propagation algorithm [2.25]). Even worse, in some cases the granularity requirements become more severe as the size of the system increases. For example, in a BAM system [2.26] of $N \times M$ neurons, storage capacity has been heuristically estimated to be around $n_p = (N \times M)^{1/4}$ [2.27], where n_p is the average maximum number of patterns that can be stored. The resolution required by the interconnection weights in this case is at least $n_p + 1$. In the chip described in this Appendix, since it is based on the ART 1 algorithm and requires only binary-valued weights, the resolution of the weights is not affected by the size nor the storage capacity of the system. This, and the non necessity of analog weights is one of the most hardware attractive features of the ART 1 algorithm.

Another consideration to take into account during the design of a hardware system is how it scales up with size and performance. We have already mentioned that some neural systems need to increase their weight resolution as they scale up. Another feature is how their size and interconnectivity scale up with pattern size or storage capacity. For an ART 1 based system, the number of neurons N in the bottom layer is the number of pixels of the patterns, the number of neurons M in the top layer is the maximum number of categories, and $N \times M$ is the number of synapses. This system scales up linearly with storage capacity (M) and input pixels (N). For a BAM system, for example, the size scales quadratically with the storage capacity and the number of pixels.

Section 2.4 will present other scaling considerations, more directly related to the hardware technique selected. In the case of an analog hardware, random and systematic errors due to fabrication process variations will appear. A neural network can usually cope very well with random errors, even if the size of the system increases. However, systematic errors may accumulate as the system increases and may render the complete network useless as it scales up. The chosen circuit technique must be either insensitive to the accumulation of systematic errors, or allow for some kind of calibration technique to overcome them.

Regarding hardware implementations of the ART 1 architecture, several attempts have been reported in the literature. Ho et al. suggested a *Type-1* implementation[3] [2.28]. Tsay and Newcomb proposed a CMOS circuit

technique that would realize a partial *Type-2* implementation [2.29]; Wunsch et al. [2.30] have built optical-based *Type-3* implementations; this work presents a CMOS VLSI *Type-3* circuit.

The next Section describes the circuit implementation of the modified ART 1 (see Appendix 1) algorithm using analog current-mode circuit design techniques.

2.2. Circuit Description

Fig. 2.1 shows the VLSI-friendly *Type-3* ART 1 algorithm as described in the previous Appendix, which has been mapped into hardware.

The operations in Fig. 2.1 that need to be implemented are the following:

- Generation of the terms T_j or "choice functions". Since z_{ij} and I_i are binary valued (0 or 1), "binary multiplication" and addition/substraction operations are required.
- Winner-Take-All (WTA) operation to select the maximum T_j term.
- Comparison of the term $\rho|\mathbf{I}|$ with $|\mathbf{I} \cap \mathbf{z}_J|$.
- Deselection of the term T_J if $\rho|\mathbf{I}| > |\mathbf{I} \cap \mathbf{z}_J|$.
- Update of weights.

The first three operations require a certain amount of precision, while the last two operations are not precise. We intended to obtain a precision between 1 and 2% (equivalent to 6-bits) for our circuit, while handling input patterns of up to 100 binary pixels. Fig. 2.2 shows a possible hardware block diagram that would physically

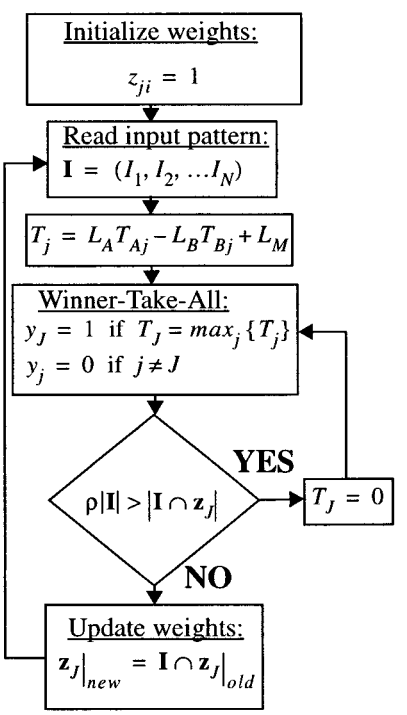


Fig. 2.1: *Type-3* implementation algorithm of the modified VLSI-friendly ART 1 architecture

3. For an explanation of the terminology *Type-1*, *Type-2*, and *Type-3* implementation refer to Appendix 1, Section 1.2.

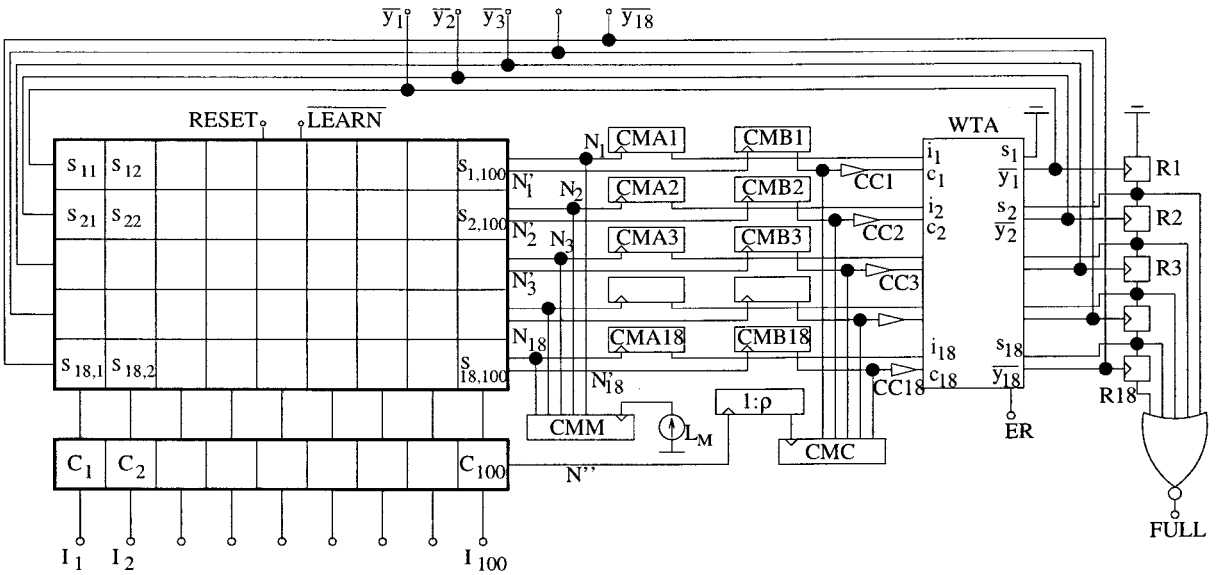


Fig. 2.2: Hardware Block Diagram for the Modified VLSI-friendly ART 1 Algorithm

implement the algorithm of Fig. 2.1. The circuit consists of an 18×100 array of *synapses* $S_{11}, S_{12}, \ldots, S_{18,100}$, a 1×100 array of *controlled current sources* $C_1, C_2, \ldots, C_{100}$, two 1×18 arrays of unity-gain current mirrors $CMA1, \ldots, CMA18, CMB1, \ldots CMB18$, a 1×18 array of current comparators $CC1, \ldots, CC18$, an 18-input WTA circuit, two 18-output unity-gain current mirrors *CMM* and *CMC*, and an adjustable-gain ($0 < \rho \leq 1$) current mirror. Registers $R1, \ldots, R18$ and the NOR gate are optional, and their function is explained later.

Each synapse receives two input signals $\bar{y}_j$ and I_i, has two global control signals RESET and $\overline{\text{LEARN}}$, stores the value of z_{ij}, and generates two output currents:

- the first goes to the input of current mirror *CMAj* and is $L_A z_{ij} I_i - L_B z_{ij}$.
- the second goes to the input of current mirror *CMBj* and is $L_A z_{ij} I_i$.

All synapses in the same row j ($S_{j1}, S_{j2}, \ldots, S_{j,100}$) share the two nodes ($N_j$ and N_j') into which the currents they generate are injected. Therefore, the input of current mirror *CMAj* receives the current

$$T_j = L_A \sum_{i=1}^{100} z_{ij} I_i - L_B \sum_{i=1}^{100} z_{ij} + L_M = L_A |\mathbf{I} \cap \mathbf{z}_j| - L_B |\mathbf{z}_j| + L_M \tag{2.4}$$

while the input of current mirror *CMBj* receives the current

$$L_A \sum_{i=1}^{100} z_{ij} I_i = L_A |\mathbf{I} \cap \mathbf{z}_j| \tag{2.5}$$

Current L_M, which is replicated 18 times by current mirror *CMM* has an arbitrary value as long as it assures that the terms T_j are positive.

Each element of the array of *controlled current sources* C_i has one input signal I_i and generates the current $L_A I_i$. All elements C_i share their output node, so that the total current they generate is $L_A |\mathbf{I}|$. This

current reaches the input of the adjustable gain ρ current mirror, and is later replicated 18 times by current mirror *CMC*.

Each of the 18 current comparators *CCj* receives the current $L_A|\mathbf{I} \cap \mathbf{z}_j| - L_A\rho|\mathbf{I}|$ and compares it against zero. If this current is positive, the output of the current comparator falls, but if the current is negative the output rises. Each current comparator *CCj* output controls input c_j of the WTA. If c_j is high the current sunk by the WTA input i_j (which is T_j) will not compete for the winning node. On the contrary, if c_j is low, input current T_j will enter the WTA competition. The outputs of the WTA $\bar{y}_j$ are all high, except for that which receives the largest $\bar{c}_j T_j$: such output, denominated $\bar{y}_J$, will fall.

Now we can describe the operation of the circuit in Fig. 2.2. All synaptic memory values z_{ij} are initially set to '1' by the RESET signal. Once the input vector $\mathbf{I}$ is activated, the 18 rows of synapses generate the currents $L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j|$ and $L_A|\mathbf{I} \cap \mathbf{z}_j|$, and the row of controlled current sources $C_1, \ldots, C_{100}$ generates the current $L_A|\mathbf{I}|$. Each current comparator *CCj* will prevent current $T_j = L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M$ from competing in the WTA if $\rho|\mathbf{I}| > |\mathbf{I} \cap \mathbf{z}_j|$. Therefore, the effective WTA inputs are $\{\bar{c}_j T_j\}$, from which the WTA chooses the maximum, making the corresponding output $\bar{y}_J$ fall. Once $\bar{y}_J$ falls, and assuming the synaptic control signal $\overline{\text{LEARN}}$ is low, all z_{iJ} values will change from '1' to 'I_i'.

Note that initially (when all $z_{ij} = 1$),

$$\bar{c}_j T_j = L_A|\mathbf{I}| - L_B N + L_M \qquad (N{=}100) \qquad \forall j \tag{2.6}$$

This means that the winner will be chosen among 18 equal competing inputs, basing the election on mismatches due to random process parameter variations of the transistors. Even after some categories are learned, there will be a number of uncommitted rows ($z_{1j} = \ldots = z_{100,j} = 1$) that generate the same competing current of eq. (2.6). The operation of a WTA circuit in which there are more than 1 equal and winning inputs becomes more difficult and in the best case, renders slower operation. To avoid these problems 18 D-registers, $R1, \ldots, R18$, might be added. Initially these registers are set to '1' so that the WTA inputs $s_2, \ldots, s_{18}$ are high. Inputs $s_1, \ldots, s_{18}$ have the same effect as inputs $c_1, \ldots, c_{18}$: if s_j is high T_j does not compete for the winner, but if s_j is low T_j enters the WTA competition. Therefore, initially only $\bar{c}_1 T_1$ competes for the winner. As soon as $\bar{y}_1$ rises once, the input of register $R1$ (which is '0') is transmitted to its output making $s_2 = 0$. Now both $\bar{c}_1 T_1$ and $\bar{c}_2 T_2$ will compete for the winner. As soon as $\bar{c}_2 T_2$ wins once, the input of register $R2$ is transmitted to its output making $s_3 = 0$. Now $\bar{c}_1 T_1$, $\bar{c}_2 T_2$, and $\bar{c}_3 T_3$ will compete, and so on. If all available F_2 nodes ($y_1, \ldots, y_{18}$) have won once, the "FULL" signal rises, advising that all F_2 nodes are storing a category. The WTA control signal "ER" enables operation of the registers.

A. Synaptic Circuit and Controlled Current Sources:

The details of a synapse S_{ij} are shown in Fig. 2.3(a). It consists of three current sources (two of value L_A and one of value L_B), a two-inverter loop (acting as a Flip-Flop), and nine MOS transistors working as switches. As can be seen in Fig. 2.3(a) each synapse generates the currents $L_A z_{ij} I_i - L_B z_{ij}$ and $L_A z_{ij} I_i$. The

RESET control signal sets z_{ij} to '1'. Learning is performed by making z_{ij} change from '1' to '0' whenever $\overline{\text{LEARN}} = 0$, $\bar{y}_j = 0$, and $I_i = 0$.

Fig. 2.3(b) shows the details of each controlled current switch C_i. If $I_i = 0$ no current is generated, while if $I_i = 1$, the current L_A is provided.

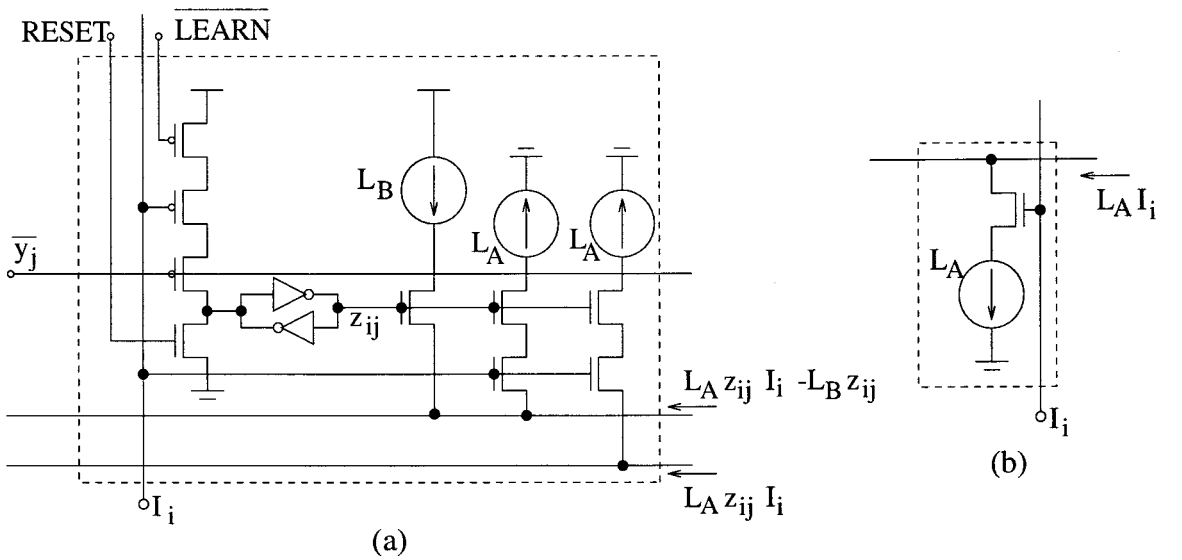


Fig. 2.3: (a) Details of Synapse Circuit S_{ij}, (b) Details of Controlled Current Source Circuit C_i

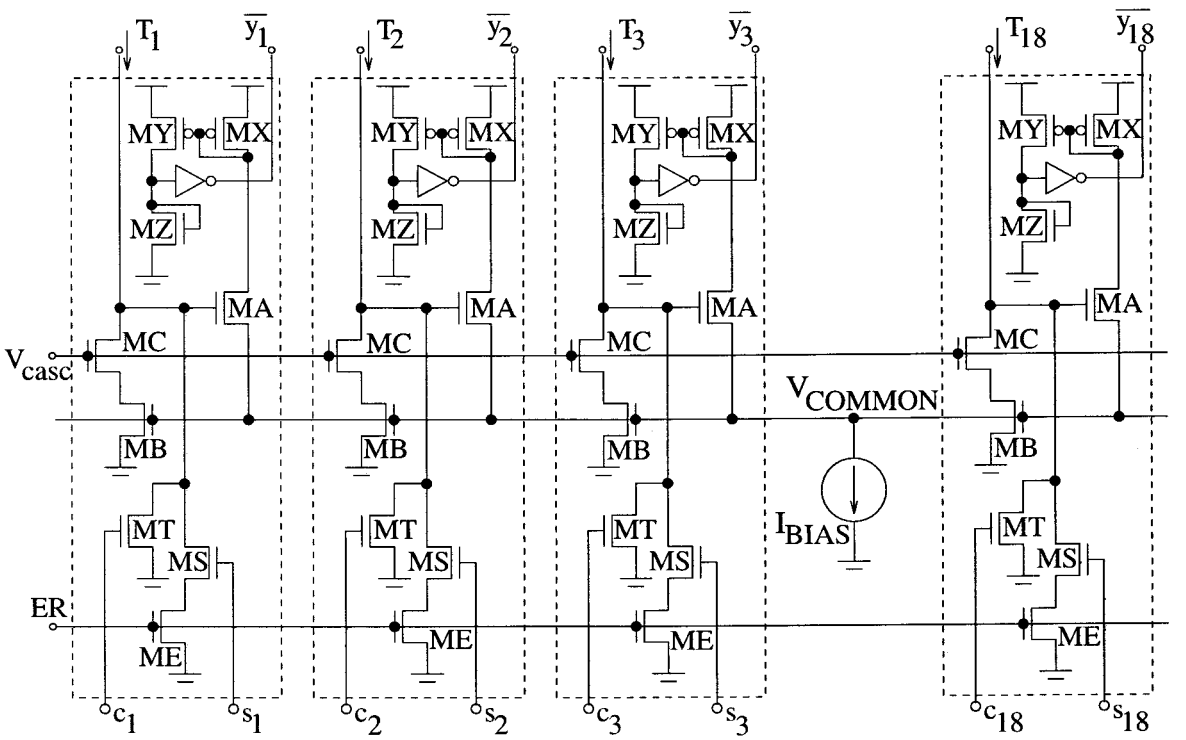


Fig. 2.4: Circuit Schematic of Winner-Take-All (WTA) Circuit

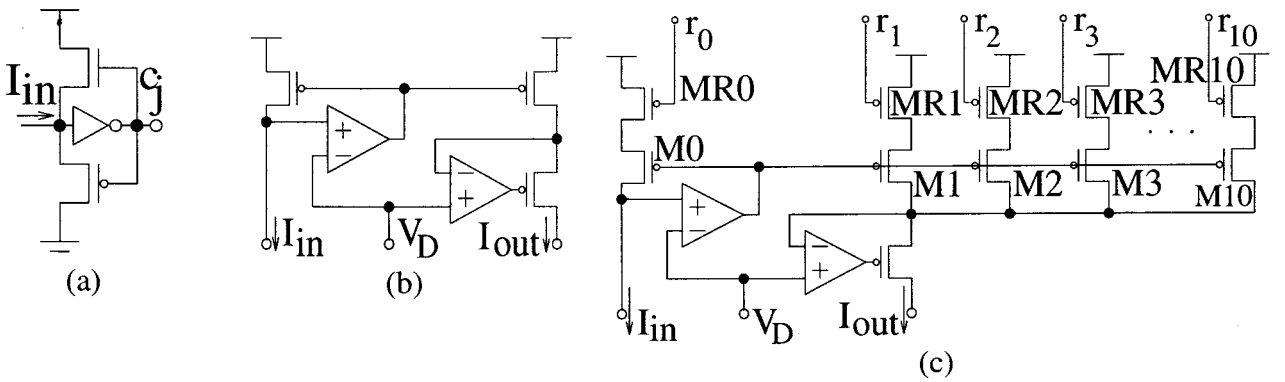


Fig. 2.5: (a) Circuit Schematic of Current Comparator. (b) Circuit Schematic of Active-Input Regulated-Cascode Current Mirror. (c) Circuit Schematic for Adjustable Gain ρ Current Mirror

B. Winner-Take-All (WTA) Circuit:

Fig. 2.4 shows the details of the WTA circuit. It is based on Lazzaro's WTA [2.31], which consists of the array of transistors *MA* and *MB*, and the current source I_{BIAS}. Transistor *MC* has been added to introduce a cascode effect and increase the gain of each cell. Transistors *MX*, *MY*, and *MZ* transform the output current into a voltage, which is then inverted to generate $\bar{y}_j$. Transistor *MT* disables the cell if c_j is high, so that the input current T_j will not compete for the winner. Transistors *MS* and *ME* have the same effect as transistor *MT*: if signals ER and s_j are high, T_j will not compete.

C. Current Comparators:

The circuit used for the current comparators is shown in Fig. 2.5(a). Such a comparator forces an input voltage approximately equal to the inverters trip voltage, has extremely high resolution (less than $1pA$), and can be extremely fast (in the order of 10-20ns for input around $10\mu A$) [2.32].

D. Current Mirrors:

Current Mirrors $CMA1, \ldots, CMA18, CMB1, \ldots, CMB18, CMM, CMC$, and the ρ-gain mirror have been laid out using common centroid layout techniques to minimize matching errors and keep the 6-bit precision of the overall system. For current mirrors $CMA1, \ldots, CMA18$ and $CMB1, \ldots, CMB18$ a special topology has been used, shown in Fig. 2.5(b) [2.33]. This topology forces a constant voltage V_D at its input node, thus producing a virtual ground in the output nodes of all synapses, which reduces channel length modulation distortion improving matching between the currents generated by all synapses. In addition, the topology of Fig. 2.5(b) presents a very wide current range with small matching errors [2.33].

The adjustable gain ρ current mirror also uses this topology, as shown in Fig. 2.5(c). Transistor *M0* has a geometry factor (W/L) 10 times larger than transistors $M1, \ldots, M10$. Transistors $MR1, \ldots, MR10$ act as switches (controlled by signals $r_1, \ldots, r_{10}$), so that the gain of the current mirror can be adjusted between $\rho = 0.0$ to $\rho = 1.0$ in steps of 0.1, while maintaining $r_0 = 0$. By making r_0 higher than 0 Volts, ρ can be fine tuned.

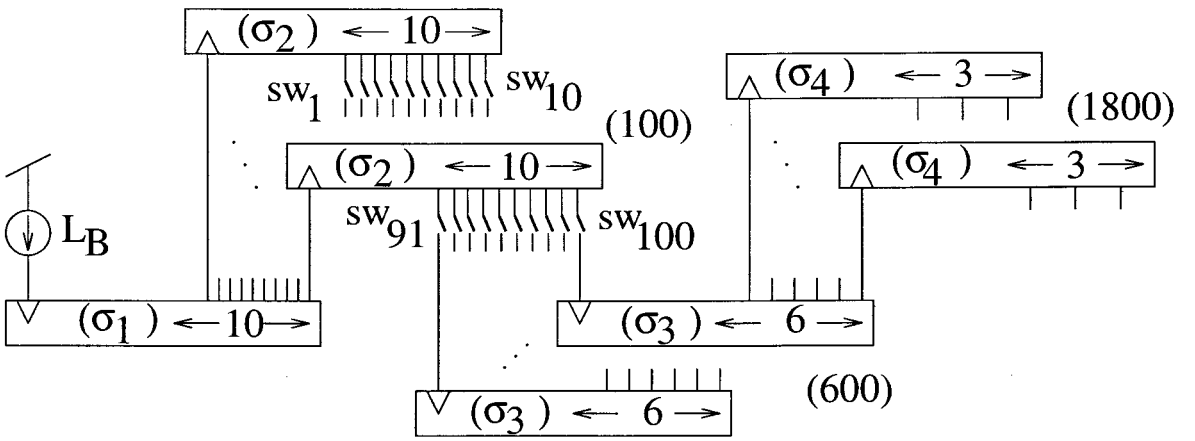


Fig. 2.6: Cascade of Current Mirrors for low Mismatching

E. Synaptic Current Sources:

The current sources L_A and L_B inside each synapse S_{ij} and controlled current sources C_i have to match within approximately 1% to keep the system 6-bit precision. There is a total of $100 \times 18 \times 2 + 100 = 3700$ L_A current sources and $100 \times 18 = 1800$ L_B current sources spread over a die area of $1cm^2$ which have to match within 1%. For such distances, number of current sources, and reasonable current values, a spread of 10% in the currents would be an optimistic estimate. However, a single current mirror, with a reduced number of outputs (like 10), a reasonable transistor size (like $40\mu m \times 40\mu m$), a moderate current (around $10\mu A$), and using common centroid layout techniques can be expected to have a mismatch error standard deviation σ_q of less than 1% [2.34]. By cascading several of these current mirrors in a tree-like fashion as is shown in Fig. 2.6 (for current sources L_B), a high number of current sources (copied from a single common reference) can be generated with a mismatch equal to

$$\sigma_{Total} = \sigma_1 + \sigma_2 + \ldots + \sigma_q \tag{2.7}$$

Each current mirror stage introduces an error σ_k. This error can be reduced by increasing the transistor areas of the current mirrors. Since the last stage q has a higher number of current mirrors, it is important to keep their area low. For previous stages the transistors can be made larger to contribute with a smaller σ_k, because they are less in number and will not contribute significantly to the total transistor area. For current sources L_A, a circuit similar to that shown in Fig. 2.6 is used. Current L_B in Fig. 2.6 (and similarly current L_A) is injected externally into the chip so that parameter $\alpha = L_A/L_B$ can be controlled.

F. Weights Read Out:

The switches sw_1 to sw_{100} of Fig. 2.6 were added to enable reading out the internally learned synaptic weights z_{ij}, and test the progress of the learning algorithm. These switches are all ON during normal operation of the system. However, for weights read-out, all except one will be OFF. The switch that is ON is selected by a decoder inside the chip, so that only column i of the synaptic array of Fig. 2.2 injects the current

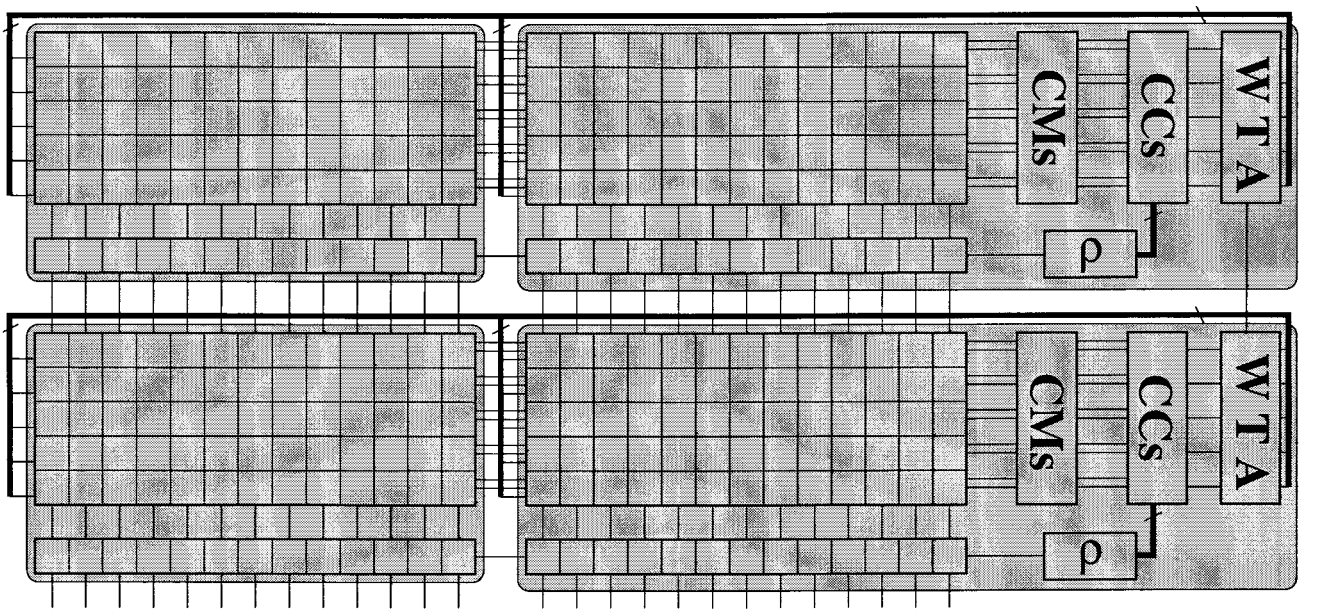


Fig. 2.7: Interchip Connectivity for Modular System Expansion

$z_{ij}L_B$ to nodes N_j. All nodes N_j can be isolated from current mirrors $CMAj$, and connected to output pads to sense the currents $z_{ij}L_B$, thus measuring the values of z_{ij}.

G. Modular System Expansibility:

The circuit of Fig. 2.2 can be expanded both horizontally, increasing the number of input patterns from 100 to $100 \times N$, and vertically increasing the number of possible categories from 18 to $18 \times M$. Fig. 2.7 shows schematically the interconnectivity between chips in the case of a 2×2 array.

Vertical expansion of the system is possible by making several chips share the input vector terminals $I_1, \ldots, I_{100}$, and node V_{COMMON} of the WTA (see Fig. 2.4). Thus, the only requirement is that V_{COMMON} be externally accessible. Horizontal expansion is directly possible by making all chips in the same row share their N_j, N'_j, and N'' nodes, and isolating all except one of them, from the current mirrors $CMA1, \ldots, CMA18$, $CMB1, \ldots, CMB18$, and the adjustable gain ρ-mirror. Also, all synapse inputs $\bar{y}_j$ must be shared.

Both vertical and horizontal expansion degrades the system performance. Vertical expansion causes degradation because the WTA becomes distributed among several chips. For the WTA of Fig. 2.4, all MA and MB transistors must match well, which is very unlikely if they are in different chips. A solution for this problem is to use a WTA topology based on current processing and replication, insensitive to inter-chip transistor mismatches [2.35], [2.36].

Horizontal expansion degrades the performance because current levels have to be changed:

- Either currents L_A and L_B are maintained the same, which makes the current mirrors $CMAj$, $CMBj$, CMM, $1{:}\rho$, CMC, the current comparators CCj, and the WTA to handle higher currents. This may cause malfunctioning due to eventual saturation in some of the blocks.

- Or currents L_A and L_B are scaled down so that the current mirrors $CMAj$, $CMBj$, CMM, $1{:}\rho$, CMC, the current comparators CCj, and the WTA handle the same current level. However, this produces an increase in mismatch between the current sources L_A and L_B.

2.3. Experimental Results

A prototype chip that contains the previous circuit description of a real-time clustering engine has been fabricated in a standard double-poly double-metal 1.6μm CMOS digital process (Eurochip ES2). The die area is $1cm^2$ and it has been mounted in a 120-pin PGA package. This chip implements an ART 1 system with 100 nodes in the F_1 layer and 18 nodes in the F_2 layer. Most of the pins are intended for test and characterization purposes. All the subcircuits in the chip can be isolated from the rest and conveniently characterized. The F_1 input vector **I**, which has 100 components, has to be loaded serially through one of the pins into a shift register. The time delay measurements reported here do not include the time for loading the shift register.

The experimental measurements provided in this Section have been divided into four parts. The first describes DC characterization results of the elements that contribute critically to the overall system precision. These elements are the WTA circuit and the synaptic current sources. The second describes time delay measurements that contribute to the global throughput time of the system. The third presents system level experimental behaviors obtained with digital test equipment (HP82000). Finally, the fourth focuses on yield and fault tolerance characterizations.

A. System Precision Characterizations:

The ART 1 chip was intended to achieve an equivalent 6-bit (~1.5% error) precision. The part of the system that is responsible for the overall precision is formed by the components that perform analog computations. These components are (see Fig. 2.2) all current sources L_A and L_B, all current mirrors $CMAj$, $CMBj$, CMM, CMC, and the ρ-mirror, the current comparators CCj, and the WTA circuit. The most critical of these components (in precision) is the WTA circuit. Current sources and current mirrors can be made to have mismatch errors below 0.2% [2.34], [2.37]-[2.39], at the expense of increasing transistors area and current, decreasing distances between matched devices, and using common centroid layout techniques [2.40]. This is feasible for current mirrors $CMAj$, $CMBj$, CMM, CMC, and the ρ-mirror, which appear in small numbers. However, the area and current level is limited for the synaptic current sources L_A and L_B, since there are many of them. Therefore, WTA and current sources L_A and L_B are the elements that limit the precision of the overall system, and their characterization results will be described next.

T_j	$10\mu A$	$100\mu A$	$1mA$
$\sigma(T_j)$	1.73%	0.86%	0.99%

Table 2.1. Precision of the WTA

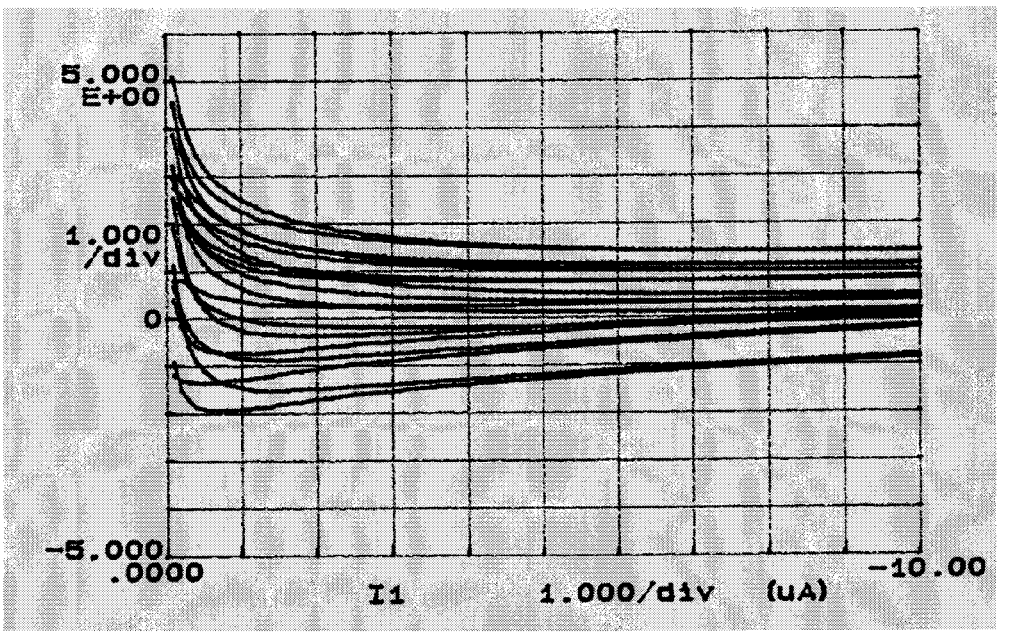


Fig. 2.8: Measured Mismatch error (in %) between 18 arbitrary L_A current sources

A.1 : WTA Precision Measurements:

L_A and L_B will have current values of 10μA or less. The maximum current a WTA input branch can receive is (see eq. (2.4)),

$$T_j|_{max} = L_M + \left[\sum_{i=1}^{100} z_{ij}(L_A I_i - L_B)\right]_{max} = L_M + 100(L_A - L_B) \tag{2.8}$$

which corresponds to the case where all z_{ij} and I_i values are equal to '1' (remember that $L_A > L_B > 0$). In our circuit the WTA was designed to handle input currents of up to $1.5mA$ for each input branch. In order to measure the precision of the WTA, all input currents except two were set to zero. Of these two inputs one was set to $100\mu A$ and the other was swept between $98\mu A$ and $102\mu A$. This will cause their corresponding output voltages $\bar{y}_j$ to indicate an interchange of winners. The transitions do not occur exactly at $100\mu A$. Moreover, the transitions change with the input branches. The standard deviation of these transitions was measured as σ=0.86μA (or 0.86%). Table 2.1 shows the standard deviation (in %) measured when the constant current is set to 10μA, 100μA, and $1mA$.

A.2 : Synaptic Current Sources Precision Measurements:

The second critical precision error source of the system is the mismatch between synaptic current sources. In our chip each of the 3700 L_A current sources and each of the 1800 L_B current sources could be isolated and independently characterized. Fig. 2.8 shows the measured mismatch error (in %) for 18 arbitrary L_A current sources when sweeping L_A between $0.1\mu A$ and $10\mu A$. As can be seen in Fig. 2.8, for currents higher than $5\mu A$ the standard deviation of the mismatch error is close to 1%. The same result is obtained for the L_B current sources.

T_1^a	T_1^b	T_2	$T_3, \ldots T_{18}$	t_{d1}	t_{d2}
$0\mu A$	$200\mu A$	$100\mu A$	0	$550ns$	$570ns$
$0\mu A$	$1mA$	$500\mu A$	0	$210ns$	$460ns$
$100\mu A$	$150\mu A$	$125\mu A$	$100\mu A$	$660ns$	$470ns$
$400\mu A$	$600\mu A$	$500\mu A$	$400\mu A$	$440ns$	$400ns$
$500\mu A$	$1.50mA$	$1.00mA$	$500\mu A$	$230ns$	$320ns$
$90\mu sA$	$110\mu A$	$100\mu A$	0	$1.12\mu s$	$1.11\mu s$
$490\mu A$	$510\mu A$	$500\mu A$	0	$1.19\mu s$	$1.06\mu s$
$990\mu A$	$1.01mA$	$1.00mA$	0	$380ns$	$920ns$

Table 2.2. Delay times of the WTA

B. Throughput Time Measurements:

For a real-time clustering device the throughput time is defined as the time needed for each input pattern to be processed. During this time the input pattern has to be classified into one of the pre-existing categories or assigned to a new one, and the pre-existing knowledge of the system has to be updated to incorporate the new information the input pattern carries. From a circuit point of view, this translates into the measurement of two delay times:

- The time needed by the WTA to select the maximum among all $\{\bar{c}_j T_j\}$.
- The time needed by the synaptic cells to change z_{ij} from its old value to $y_j I_i z_{ij}$

B.1 : WTA Delay Measurements:

The delay introduced by the WTA depends on the current level present in the competing input branches. This current level will depend on the values chosen for L_A, L_B, and L_M, as well as on the input pattern $\mathbf{I}$ and all internal weights $\mathbf{z}_j$. To keep the presentation simple, delay times will be given as a function of T_j values directly. Table 2.2 shows the measured delay times when T_1 changes from T_1^a to T_1^b, and T_2 to T_{18} have the values given in the table. t_{d1} is the time needed by category y_1 to win when T_1 switches from T_1^a to T_1^b, and t_{d2} is the time spent by category y_2 in winning when T_1 decreases from T_1^b to T_1^a. As can be seen, this delay is always below $1.2\mu s$.

For the cases when the vigilance criterion is not directly satisfied and hence comparators CCj cut some of the T_j currents, an additional delay is observed. This extra delay has been measured to be less than $400ns$ for the worst cases. Therefore, the time needed until the WTA selects the maximum among all $\{\bar{c}_j T_j\}$ is less than $1.2\mu s + 0.4\mu s = 1.6\mu s$.

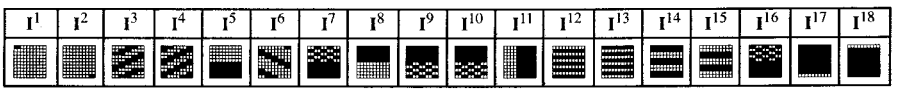


Fig. 2.9: Set of Input Patterns

B.2 : Learning Time:

After a delay of $1.6\mu s$ (so that the WTA can settle), the learn signal $\overline{LEARN}$ (see Fig. 2.2) is enabled during a time t_{LEARN}. To measure the minimum t_{LEARN} time required, this time was set to a specific value during a training/learning trial, and it was checked that the weights had been updated properly. By progressively decreasing t_{LEARN} until some of the weights did not update correctly, it was found that the minimum t_{LEARN} time for proper operation was $190ns$. By setting t_{LEARN} to $200ns$ and allowing the WTA a delay of $1.6\mu s$, the total throughput time of the ART 1 chip is established as $1.8\mu s$.

B.3 : Comparison with Digital Neural Processors:

A digital chip with a feedforward speed of a connections per second, a learning speed of b connection updates per second, and a WTA section with a delay of c seconds must satisfy the following equation to achieve a throughput time of $1.8\mu s$ when emulating the ART 1 algorithm of Fig. 2.1(c):

$$\frac{3700}{a} + \frac{100}{b} + c = 1.8\mu s \tag{2.9}$$

Note that there are 100 synapse weights z_{ij} to update for each pattern presentation, and 3700 feed-forward connections: 1800 connections to generate all $T_j = L_A|\mathbf{I} \cap \mathbf{z}_j| - L_B|\mathbf{z}_j| + L_M$, 1800 connections to generate $L_A|\mathbf{I} \cap \mathbf{z}_j|$, and 100 connections to generate $L_A|\mathbf{I}|$.

Assuming $c = 100ns$, and $a = b$, eq. (2.9) results in a processing speed of $a = b = 2.2 \times 10^9$ connections/s and connection-updates/s. A digital neural processor would require such figures of merit to equal the processing time of the analog ART 1 chip presented in this work. Therefore, this "approximate reasoning" makes us conclude that our chip has an equivalent computing power of $a + b = 4.4 \times 10^9$ connections/s plus connection-updates/s.

C. System Level Performance:

Although the internal processing of the chip is analog in nature, its input (I_i) and output ($\overline{y_j}$) are binary valued. Therefore, the system level behavior of the chip can be tested using conventional digital test equipment. In our case we used the HP82000 IC Evaluation System.

An arbitrary set of 100-bit input patterns $\{\mathbf{I}^k\}$ was chosen, shown in Fig. 2.9. A typical clustering sequence is shown in Fig. 2.10, for $\rho = 0.7$ and $\alpha = L_A/L_B = 1.05$. The first column indicates the input pattern $\mathbf{I}^k$ that is fed to the F_1 layer. The other 18 squares (10×10 pixels) in each row represent each of the internal $\mathbf{z}_j$ vectors after learning is finished. The vertical bars to the right of some $\mathbf{z}_j$ squares indicate that these categories won the WTA competition while satisfying the vigilance criterion. Therefore, such categories correspond to $\mathbf{z}_J$, and these are the only ones that are updated for that input pattern $\mathbf{I}^k$ presentation. The figure shows only two iterations of input patterns presentation, because no change in weights were observed after these. The last row of weights $\mathbf{z}_j$ indicates the resulting categorization of the input patterns. The numbers

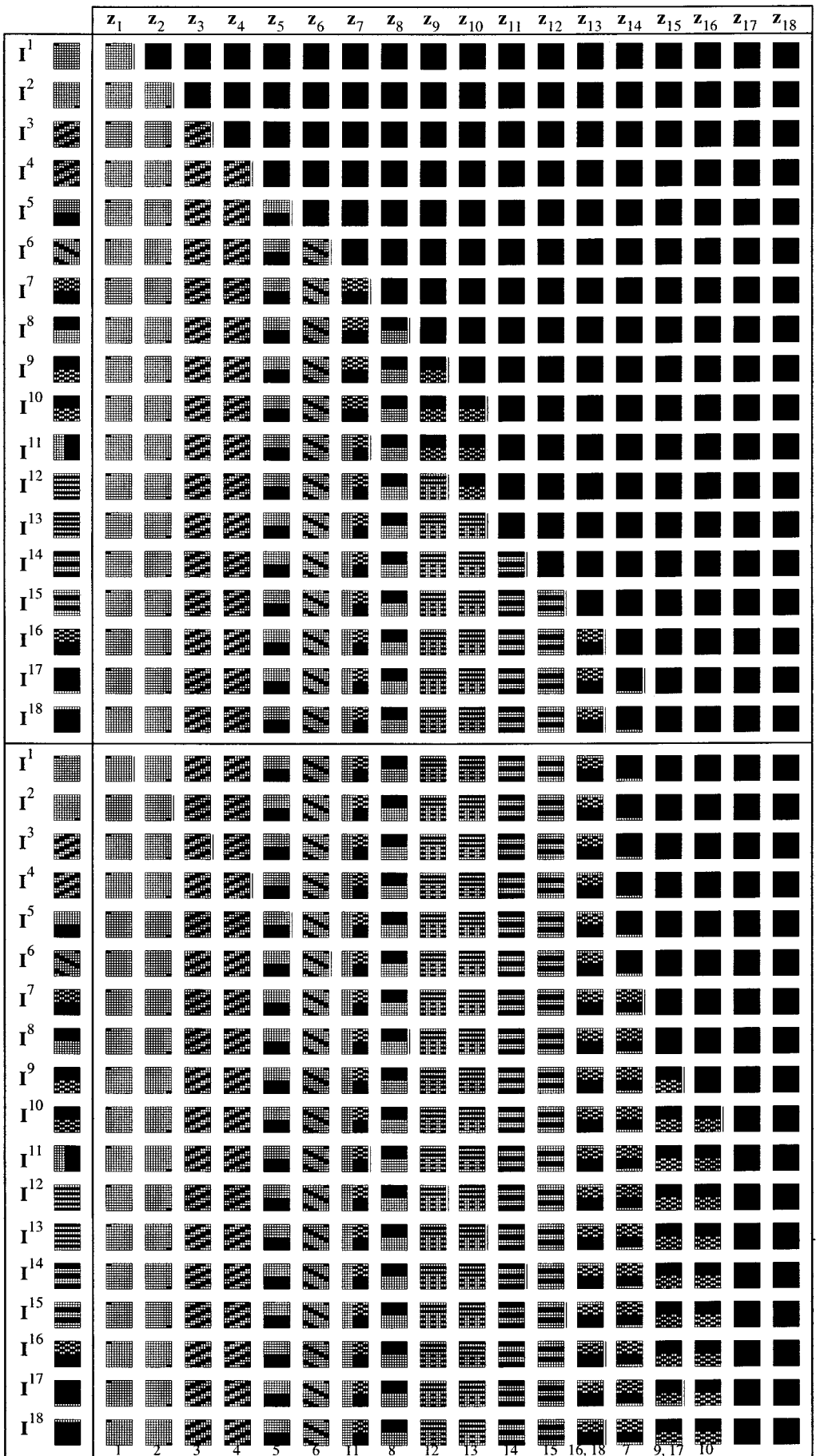


Fig. 2.10: Clustering Sequence for ρ=0.7 and $\alpha=L_A/L_B$=1.05

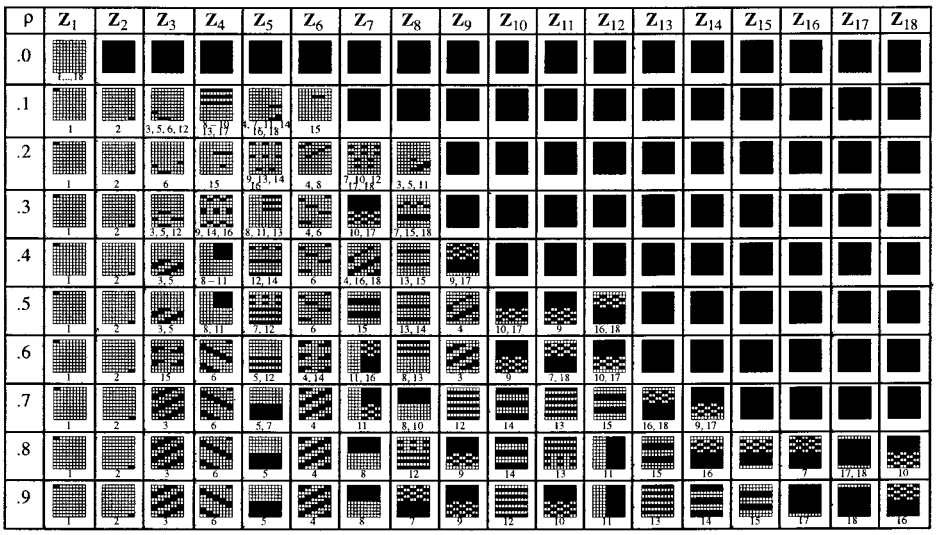


Fig. 2.11: Categorization of the input patterns for L_A=3.2μA, L_B=3.0μA, L_M=400μA, and different values of ρ

below each category indicate the input patterns that have been clustered into this category. In the following figures we will show only this last row of learned patterns together with the pattern numbers that have been clustered into each category.

Fig. 2.11 shows the categorizations that result when tuning the vigilance parameter ρ to different values while the currents were set to $L_A = 3.2\mu A$, $L_B = 3.0\mu A$, and $L_M = 400\mu A$ ($\alpha = L_A/L_B = 1.07$). Note that below some categories there is no number. This is a known ART 1 behavior: during the clustering process some categories might be created that will not represent any of the training patterns. In Fig. 2.12 the vigilance parameter is maintained constant at $\rho = 0$, while α changes from 1.07 to 50. For a more detailed explanation on how and why the clustering behavior depends on ρ and α see references [2.3] and [2.4], or other ART 1 theoretical papers [2.2], [2.41].

D. Yield and Fault Tolerance:

A total of 30 chips (numbered 1 through 30 in Table 2.3 and Fig. 2.13) were fabricated. For each chip every subcircuit was independently tested and its proper operation verified; 14 different faults were identified. Table 2.3 indicates the faults detected for each of the 30 chips. The faults have been denoted from **F1** to **F14**, and are separated into two groups:

- *Catastrophic Faults (digital sense)* are those clearly originated by a short or open circuit failure. These faults are **F1**, ... **F8**. This kind of faults would produce a failure in a digital circuit.
- *Non-Catastrophic Faults (digital sense)* are those that produce a large deviation from the nominal behavior, too large to be explained by random process parameter variations. These faults are **F9**, ... **F14**. This kind of faults would probably not produce a catastrophic failure in a digital circuit, but be responsible for significant delay times degradations.

	Catastrophic Faults (digital sense)								Non-Catastrophic Faults (digital sense)					
chip #	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**	**F11**	**F12**	**F13**	**F14**
1					X						X			
2		X					X		X	X		X	X	
3									X	X		X		X
4		X							X	X		X		
5	X	X							X					
6			X	X					X	X	X			
7			X						X					
8									X					
9		X		X					X	X				
10			X	X					X	X				
11	X													
12				X						X				
13	X	X	X						X	X				
14		X							X					
15		X	X						X	X				
16			X						X					
17	X		X						X					
18									X	X		X		
19										X				
20		X	X	X					X	X		X		
21	X								X					
22									X			X		
23	X								X					
24				X						X				
25														
26									X					
27	X	X	X						X					
28														
29	X													
30									X	X		X		

Table 2.3. Fault Characterizations of the 30 ART 1 Chip Samples. Dark Shades: Sample with Catastrophic Fault; Light Shade: Sample with no Catastrophic Fault but with non Catastrophic Fault; no Shade: Sample with no Fault.

Table 2.4 describes the subcircuits where the faults of Table 2.3 were found. Note that the most frequent faults are **F2/F9** and **F3/F10**, which are failures in some current sources L_A or L_B, and these current sources occupy a significant percentage of the total die area. Fault **F1** is a fault in the shift register that loads the input vector $\mathbf{I}^k$. Fault **F2** is a fault in the WTA circuit. Therefore, chips with an **F1** or **F2** fault could not be tested for system level operation. Faults **F3** and **F9** are faults detected in the same subcircuits of the chip, with **F3** being catastrophic and **F9** non-catastrophic. The same is valid for **F4** and **F10**, **F5** and **F11**, and so on until **F8** and **F14**.

Note that only 2 of the 30 chips (6.7%) are completely fault-free. According to the simplified expression for the yield performance as a function of die area Ω and process defects density ρ_D [2.42],

$$yield(\%) = 100e^{-\rho_D \Omega} \tag{2.10}$$

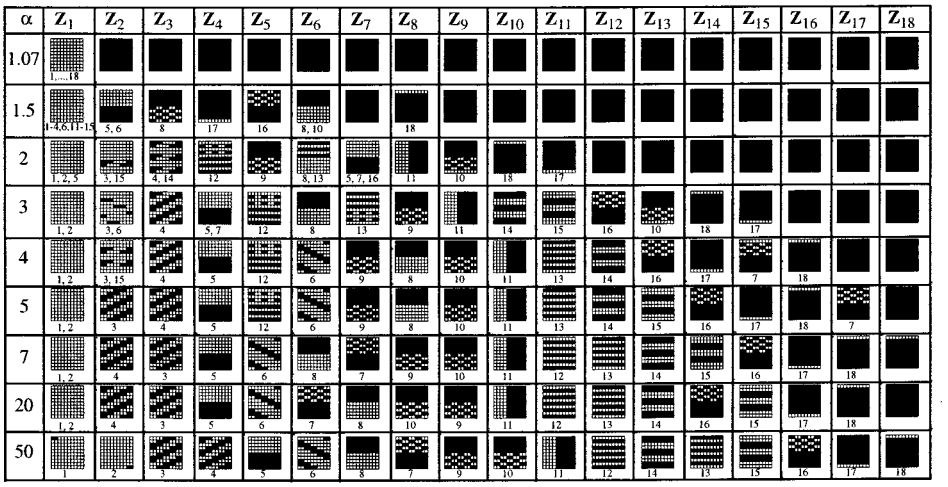


Fig. 2.12: Categorization of the input patterns for ρ=0 and different values of α

F1	non-operative shift register for loading $\mathbf{I}^k$
F2	non-operative WTA circuit
F3/F9	fault in a current source L_A
F4/F10	fault in a current source L_B
F5/F11	fault in vigilance parameter ρ current mirror
F6/F12	fault in current mirror *CMM*
F7/F13	fault in current mirrors *CMAj* or *CMBj*
F8/F14	fault in current mirror *CMC*

Table 2.4. Description of Faults

this requires a process defect density[4] of $\rho_D = 3.2cm^{-1}$. On the other hand, ignoring the non-catastrophic faults yields 9 out of 30 chips (30%). According to eq. (2.10) such a yield would be predicted if the process defect density is $\rho'_D = 1.4cm^{-1}$.

Even though the yield is quite low, many of the faulty samples were still operative. This is due to the fault tolerant nature of the neural algorithms in general [2.43]-[2.46], and the ART 1 algorithm in particular. Looking at Table 2.3 we can see that there are 16 chips that have an operative shift register and WTA circuit. We performed system level operation tests on these chips to verify if they would be able to form clusters of the input data, and verified that 12 of these 16 chips were able to do so. Moreover, 6 (among which were the two completely fault-free chips) behaved exactly identically. The resulting clustering behavior of these 12 chips is depicted in Fig. 2.13 for $\rho = 0.5$ and $\alpha = 1.07$.

4. The effective die area is $\Omega = (0.92cm)^2$ to account for a $400\mu m$ width pad ring.

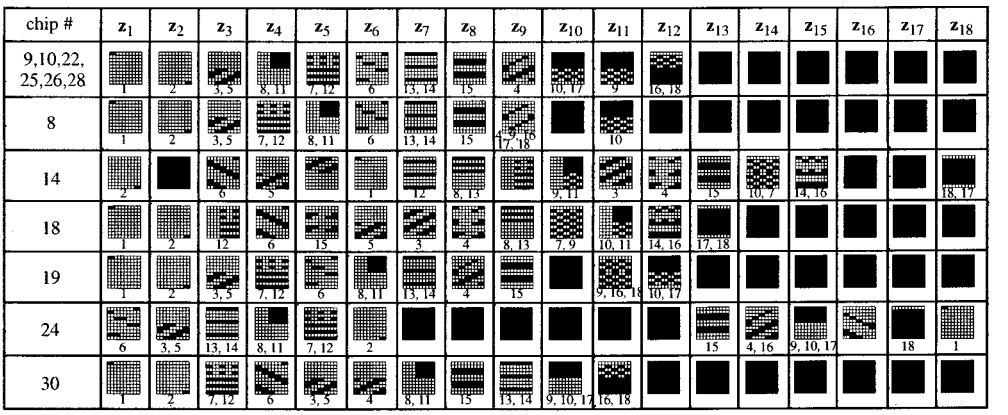


Fig. 2.13: Categorization of the input patterns performed by operative samples

2.4. Further Enhancements

The chip described here is the first prototype designed by the authors for real-time clustering. As such, the design focused on testability and full characterization possibilities, instead of maximizing speed and yield, for example.

To make this chip an industry ready prototype, several trivial modifications should be introduced:

- First, substitute the serially loaded shift register that holds the input pattern $\mathbf{I}^k$, with some kind of parallel loading mechanism (using either electrical or optical data acquisition techniques).
- Use some simple yield enhancement technique. Looking at Table 2.3 and Table 2.4 we can see that most of the failures are due to faults in the synaptic current sources. A simple yield enhancement technique would be to add a number of spare columns of synapses, some of which would substitute faulty columns of synapses.
- Add a handshaking mechanism that would allow the chip to communicate with the outside circuitry. Thus, when the WTA produces a fast response (which, by the way, is most of the times), the outside circuitry need not wait for the worst case WTA delay.

Other, less trivial, enhancements that should be addressed relate to the high area and current consumption of the synaptic current sources L_A and L_B. One possibility would be to use UV-activated floating-gate-calibrated [2.47]-[2.50] current sources, instead of the tree-like structure of Fig. 2.6. In principle, it should be possible to use one single calibrated MOS transistor per synaptic current source. This transistor, which can be close to minimum size, does not have to drive a large current either. Calibration errors of 0.2% have been reported for currents of $200nA$ [2.50]. Using a scheme like this significantly reduces the current and silicon area consumption per synapse, allowing a much higher number of synapses per chip and thus boosting the performance of the chip significantly.

Other considerations relate to the question of how this chip would scale up with size. What would be the practical limitations? Usually a strong limitation when scaling up analog neural hardware is how systematic offsets accumulate. A common circuit technique for analog neural VLSI is the use of transconductors [2.23],

[2.51]. Connecting many of them in parallel results in addition of their systematic offset components. If the size of the system is sufficiently large, this total offset can drive the system out of working range[5]. For our circuit the accumulation of systematic offsets of the synaptic current sources is not a problem. Note that the total currents T_j (which certainly include a common systematic offset) will compete in a WTA circuit, and the maximum among all $\{T_j\}$ is the same regardless of the presence or not of a common offset component.

A real scaling limitation for the circuit technique used in our chip is the following. The smallest current per synaptic current source is limited by the precision we want to achieve (even when using UV-activated floating-gate calibration techniques). Therefore, the maximum number of synapses that can be put into the same chip will be limited by the maximum power dissipation allowed by the package for a given precision. This implies a trade-off between precision and size.

Another problem that might arise when the number of nodes in the F_2 layer (maximum number of categories) becomes significantly large, is that the WTA circuit might not be able to detect the maximum among a large number of close-to-maximum inputs. At that point, one might reconsider if it is necessary to have an F_2 layer that provides one (and only one) winner, instead of an F_2 layer that provides a "bubble" of winners [2.52], [2.53].

A different way of system growth is to assemble different ART 1 subsystems to perform supervised clustering tasks [2.54], or to combine ART cells hierarchically for higher level knowledge processing [2.55], [2.56].

2.5. References

[2.1] T. Serrano-Gotarredona and B. Linares-Barranco, "A Real-Time Clustering Microchip Neural Engine," *IEEE Transaction on VLSI Systems*, 1996.

[2.2] G. A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer Vision, Graphics, and Image Processing*, vol. 37, pp. 54-115, 1987.

[2.3] T. Serrano-Gotarredona and B. Linares-Barranco, "A VLSI-friendly 'Fast-Learning' ART 1 Algorithm," *Proceedings of the 1995 World Congress on Neural Networks,* Washington DC, vol. I, pp. 27-30, 1995.

[2.4] T. Serrano-Gotarredona and B. Linares-Barranco, "A Modified ART 1 Algorithm more suitable for VLSI Implementations," *Neural Networks*, 1996.

[2.5] M. Griffin, G. Tahara, K. Knorpp, and W. Riley, "An 11-million Transistor Neural Network Execution Engine," *Proc. of the 1991 IEEE Int. Conf. on Solid-State Circuits,* 1991, pp. 180-181.

[2.6] M. Yasunaga, N. Masuda, M. Yagyu, M. Asai, K. Shibata, M. Ooyama, M. Yamada, T. Sakaguchi, and M. Hashimoto, "A Self-Learning Neural Network Composed of 1152 Digital Neurons in Wafer-Scale LSIs," *Proc. of the 1991 Int. Joint Conf. on Neural Networks,* Seattle, Washington, July 1991, pp. 1844-1849.

[2.7] N. Mauduit, M. Duranton, J. Gobert, and J. A. Sirat, "Lneuro 1.0: A Piece of Hardware LEGO for building Neural Network Systems," *IEEE Trans. on Neural Networks,* vol. 3, No. 3, May 1992, pp. 414-422.

[2.8] A. J. De Groot and S. R. Parker, "Systolic Implementation of Neural Networks," *SPIE High Speed Computing II*, vol. 1058, pp. 182-190, Los Angeles, California, 1989.

[2.9] S. Jones, K. Sammut, Ch. Nielsen, and J. Staunstrup, "Toroidal Neural Network: Architecture and Processor Granularity Issues," in *VLSI Design of Neural Networks*, U. Ramacher and U. Rueckert (Eds.), pp. 229-254, Kluwer Academic Publishers, Dordrecht, Netherlands, 1991.

[2.10] R. W. Means and L. Lisenbee, "Floating-point SIMD Neurocomputer Array Processor," *paper distributed by HNC Inc.,* 1993.

5. In this case a global offset calibration technique can be used to overcome this problem.

[2.11] U. Ramacher, J. Beichter, W. Raab, J. Anlauf, N. Bruels, U. Hachmann, and M. Wesseling, "Design of a 1st Generation Neurocomputer," in *VLSI Design of Neural Networks*, U. Ramacher and U. Rueckert (Eds.), pp. 271-310, Kluwer Academic Publishers, Dordrecht, Netherlands, 1991.

[2.12] M. A. Viredaz, C. Lehmann, F. Blayo, and P. Ienne, "MANTRA: A Multi-Model Neural-Network Computer," *Proc. of the 3rd Int. Workshop on VLSI for Neural Networks and Artificial Intelligence,* Oxford, September 1992.

[2.13] J. H. Chung, H. Yoon, and S. R. Maeng, "A Systolic Array Exploiting the Inherent Parallelisms of Artificial Neural Networks," *Microprocessing and Microprogramming,* vol. 33, pp. 145-159, 1992.

[2.14] T. Kohonen, *Self-Organization and Associative Memory,* 3rd ed., Berlin, Germany: Springer-Verlag, 1989.

[2.15] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York: Plenum, 1981.

[2.16] R. Duda and P. Hart, *Pattern Classification and Scene Analysis,* New York: Wiley, 1973.

[2.17] J. Hartigan, *Clustering Algorithms*, New York: Wiley, 1975.

[2.18] R. Dubes and A. Jain, *Algorithms that Cluster Data*, Englewood Cliffs, NJ: Prentice Hall, 1988.

[2.19] Y. H. Pao, *Adaptive Recognition and Neural Networks,* Reading, MA: Addison Wesley, 1989.

[2.20] A. Rodríguez-Vázquez, S. Espejo, R. Domínguez-Castro, J. L. Huertas, and E. Sánchez-Sinencio, "Current-Mode Techniques for the Implementation of Continuous- and Discrete-Time Cellular Neural Networks," *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 40, No. 3, March 1993, pp. 132-146.

[2.21] A. Rodríguez-Vázquez and M. Delgado-Restituto, "Generation of Chaotic Signals using Current-Mode Techniques," *Journal of Intelligent and Fuzzy Systems,* vol. 2, No. 1, pp. 15-37, 1994.

[2.22] H. Oh and F. M. A. Salam, "Analog CMOS Implementation of Neural Network for Adaptive Signal Processing," *Proc. of the 1994 IEEE Int. Symp. on Circuits and Systems (ISCAS'94),* London, 1994, pp. 503-506.

[2.23] B. Linares-Barranco, E. Sánchez-Sinencio, A. Rodríguez-Vázquez, and J. L. Huertas, "A CMOS Analog Adaptive BAM with On-Chip Learning and Weight Refreshing," *IEEE Trans. on Neural Networks,* vol. 4, No. 3, May 1993, pp. 445-455.

[2.24] E. Keulen, S. Colak, H. Withagen, and H. Hegt, "Neural Network Hardware Performance Criteria," *Proc. of the 1994 Int. Conf. on Neural Networks (ICNN'94)*, Orlando, 1994, pp. 1885-1888.

[2.25] M. Riedmiller, "Advanced Supervised Learning in Multi-layer Perceptrons - From Backpropagation to Adaptive Learning Algorithms," *Int. Journal of Computer Standards and Interfaces* (Special Issue on Neural Networks), (5), 1994.

[2.26] B. Kosko, "Adaptive Bidirectional Associative Memories," *Applied Optics,* vol. 26, pp. 4947-4960, December, 1987.

[2.27] Y. F. Wand, J. B. Cruz, and J. H. Mulligan, "On Multiple Training for Bidirectional Associative Memory," *IEEE Trans. on Neural Networks,* vol. 1, September 1990, pp. 275-276.

[2.28] C. S. Ho, J. J. Liou, M. Georgiopoulos, G. L. Heileman, and C. Christodoulou, "Analogue Circuit Design and Implementation of an Adaptive Resonance Theory (ART) Neural Network Architecture," *Int. Journal on Electronics*, vol. 76, No. 2, pp. 271-291, 1994.

[2.29] S. W. Tsay and R. W. Newcomb, "VLSI Implementation of ART 1 Memories," *IEEE Transactions on Neural Networks*, vol. 2, No. 2, pp. 214-221, March 1991.

[2.30] D. C. Wunsch II, T. P. Caudell, C. D. Capps, R. J. Marks II, and R. A. Falk, "An Optoelectronic Implementation of the Adaptive Resonance Neural Network," *IEEE Transactions on Neural Networks*, vol. 4, No. 4, pp. 673-684, July 1993.

[2.31] J. Lazzaro, R. Ryckebush, M. A. Mahowald, and C. Mead, "Winner-Take-All Networks of O(n) Complexity," in *Advances in Neural Information Processing Systems,* vol. 1, D. S. Touretzky (Ed.), Los Altos, CA: Morgan Kaufmann, 1989, pp. 703-711.

[2.32] A. Rodríguez-Vázquez, R. Domínguez-Castro, F. Medeiro and M. Delagdo-Restituto, "High Resolution CMOS Current Comparators: Design and Applications to Current-Mode Function Generation," *Analog Integrated Circuits and Signal Processing*, Kluwer Academic Publishers, 7, pp. 149-165, 1995.

[2.33] T. Serrano and B. Linares-Barranco, "The Active-Input Regulated-Cascode Current Mirror," *IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications,* vol. 41, No. 6, June 1994, pp. 464-467.

[2.34] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching Properties of MOS Transistors," *IEEE Journal of Solid-State Circuits,* vol. 24, October 1989, pp. 1433-1440.

[2.35] T. Serrano and B. Linares-Barranco, "A Modular Current-Mode High-Precision Winner-Take-All Circuit," *Proc. of the 1994 Int. Symposium on Circuits and Systems,* London, 1994, vol. 5, pp. 557-560.

[2.36] T. Serrano and B. Linares-Barranco, "A Modular Current-Mode High-Precision Winner-Take-All Circuit," *IEEE Trans. on Circuits and Systems II*, vol. 42, No. 2, pp. 132-134, February 1995.

[2.37] J. B. Shyu, G. C. Temes, and F. Krummenacher, "Random Error Effects in Matched MOS Capacitors and Current Sources," *IEEE Journal Solid-State Circuits.*, vol. SC-19, No. 6, pp. 948-955, December 1984.

[2.38] K. R. Lakshmikumar, R. A. Hadaway, and M. A. Copeland, "Characterization and Modeling of Mismatch in MOS Transistors for Precision Analog Design, " *IEEE Journal Solid-State Circuits*, vol. SC-21, No. 6, pp.1057-1066, December 1986.

[2.39] C. Michael and M. Ismail, "Statistical Modeling of Device Mismatch for Analog MOS Integrated Circuits," *IEEE Journal Solid-State Circuits.*, vol. 27, No. 2, pp. 154-166, February 1992.

[2.40] P. E. Allen and D. R. Holberg, *CMOS Analog Design*, Holt Rinehart and Winston Inc., New york, 1987.

[2.41] B. Moore, "ART 1 and Pattern Clustering," in *Proceedings of the 1988 Connectionist Summer School,* D. S. Touretzky, G. Hinton, and T. Sejnowski (Eds.), pp. 174-185, San Mateo, CA, 1989. Morgan Kaufmann.

[2.42] N. R. Strader and J. C. Harden, "Architectural Yield Optimization," in *Wafer Scale Integration*, E. E. Swartzlander, Jr. (Ed.), pp. 57-118, Kluwer Academic Publishers, Boston, 1989.

[2.43] L. C. Chu, "Fault-tolerant Model of Neural Computing," *IEEE Int. Conf. on Computer Design: VLSI in Computers and Processors*, pp. 122-125, 1991.

[2.44] C. Neti, M. H. Schneider, and E. D. Young, "Maximally Fault Tolerant Neural Networks," *IEEE Trans. on Neural Networks,* vol. 3, No. 1, pp. 14-23, January 1992.

[2.45] J. H. Kim, C. Lursinsap, and S. Park, "Fault-Tolerant Artificial Neural Nertworks," *Int. Joint Conf. on Neural Networks,* (IJCNN'91) Seattle, vol. 2, pp. 951, 1991.

[2.46] T. Petsche, "Trellis Codes, Receptive Fields, and Fault Tolerant, Self-Reparing Neural Networks," in *Machine Learning: From Theory to Applications*, Cooperative Research at Siemens and MIT, S. J. Hanson, W. Remmele, and R. L. Rivest (Eds.), Springer-Verlag, Berlin, Germany, pp. 241-268, 1993.

[2.47] D. A. Kerns, *Experiments in Very Large-Scale Analog Computations,* PhD Thesis, California Institute of Technology, Pasadena, California, 1993.

[2.48] G. Cauwenbergs, C. F. Neugebauer, and A. Yariv, "Analysis and Verification of an Analog VLSI Incremental Outer-Product Learning System," *IEEE Trans. on Neural Networks*, vol. 3, No. 3, pp. 488-497, May 1992.

[2.49] K. Yang and A. G. Andreou, "The Multiple Input Floating Gate MOS Differential Amplifier: An Analog Computational Building Block," *Proc. of the 1994 Int. Symp. on Circuits and Systems (ISCAS'94)*, London, pp. 37-40, 1994.

[2.50] H. Miwa, K. Yang, P. O. Pouliquen, N. Kumar, and A. G. Andreou, "Storage Enhancement Techniques for Digital Memory Based, Analog Computation Engines," *Proc. of the 1994 Int. Symp. on Circuits and Systems (ISCAS'94)*, London, pp. 45-48, 1994.

[2.51] B. W. Lee and B. J. Sheu, *Hardware Annealing in Analog VLSI Neurocomputing*, Boston: Kluwer Academic Publishers, 1991.

[2.52] S. Grossberg, "Nonlinear Neural Networks: Principles, Mechanisms, and Architectures," *Neural Networks*, vol. 1, pp. 17-61, 1988.

[2.53] T. Kohonen, *Self-Organization and Associative Memory*, Springer Verlag, New York, 1984.

[2.54] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network," *Neural Networks,* vol. 4, pp. 565-588, 1991.

[2.55] S. Grossberg, *Neural Networks and Natural Intelligence*, MIT Press, 1989.

[2.56] G. A. Carpenter and S. Grossberg, *Pattern Recognition by Self-Organizing Neural Networks*, MIT Press, 1991.

Appendix 3: A High-Precision Current-Mode WTA-MAX Circuit with Multi-Chip Capability

3.1. Introduction

Winner-Take-All (or Looser-Take-All) and MAX (or MIN) circuits are often fundamental building blocks in neural and/or fuzzy hardware systems [3.3]-[3.5]. Given a set of M external inputs (T_1, T_2, ..., T_j, ...T_M), their operation consists in determining which input j presents the largest (or smallest) value, or what is this maximum (or minimum) value, respectively. If a Winner-Take-All (WTA) or MAX circuit is available, a Looser-Take-All (LTA) or MIN circuit is obtained by simply inverting the input ($-T_1$, $-T_2$, ..., $-T_j$, ..., $-T_M$)[1]. Hence, this Appendix will only concentrate on WTA and MAX circuits.

In literature, the physical implementation of these systems has been tackled through two main approaches:

a) Systems of $O(M^2)$ complexity: their connectivity increases quadratically with the number of inputs [3.6]-[3.10].

b) Systems of $O(M)$ complexity: their connectivity increases linearly with the number of inputs [3.11], [3.12].

In a system of $O(M^2)$ complexity, as shown in Fig. 3.1(a), there is one cell per input; each cell has an inhibitory connection (black triangle) to the rest of the cells and an excitatory connection (white triangle) to itself. Therefore, the system has M^2 connections. Each cell j receives an external input T_j. The cell that receives the maximum input will turn all other cells OFF and will remain ON. If the system is a Winner-Take-All (WTA) circuit, each cell has a binary output that indicates whether the cell is ON or OFF. In a MAX circuit the winning cell will copy its input to a common output.

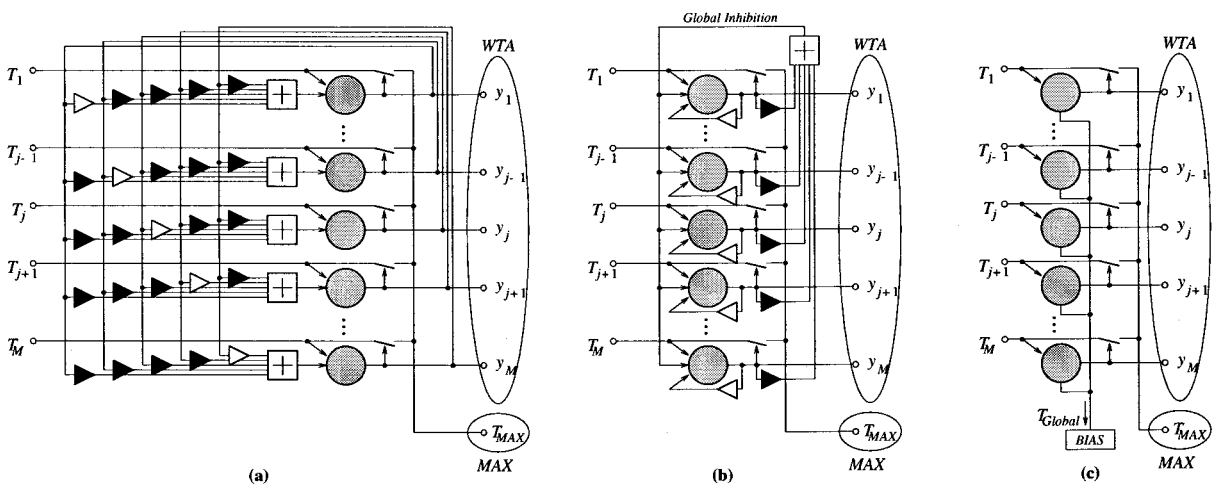


Fig. 3.1: WTA topologies. (a) WTA of $O(M^2)$ complexity, (b) transformation to $O(M)$ complexity, (c) typical topology of $O(M)$ WTA hardware implementation.

1. Optionally, a common offset term may be added.

This Appendix is a merged and amplified version of papers [3.1] and [3.2].

Under some circumstances[2] it is possible to convert the $O(M^2)$ topology of Fig. 3.1(a) into an $O(M)$ one, as shown in Fig. 3.1(b). In these cases, a global inhibition term is computed. Each cell contributes to this global inhibition, and each cell receives the same global inhibition. Note that now, each cell contributes to inhibit itself. Consequently, the excitatory connection that each cell has to itself must be increased to compensate for this fact.

Typical $O(M)$ WTA circuits reported in literature [3.11], [3.12] correspond to the topology shown in Fig. 3.1(c). In such circuits there are also M cells, each receiving an external input T_j. Each cell connects to a common node, through which a global property (for example, a current) is shared between all cells. The amount of that global property taken by each cell depends (nonlinearly) on how much its input T_j deviates from an "average" of all inputs. Usually this "average" is not an exact linear average, but is somehow nonlinearly dependent on all inputs. The cell with the maximum input T_j takes most (or all) of the common global property leaving the rest with little or nothing. Due to the way this global property is shared and how the "average" is computed, the operation of these circuits relies on the matching of transistor threshold voltages of an array of transistors [3.11], or other transistor parameters (like in [3.12] where the operation also relies on the matching of parameter λ of the transistor array). The number of transistors in the array equals, at least, the number of inputs M of the system. If the WTA or MAX circuit has such a large number of inputs so that it must be distributed among different chips, the matching of threshold voltages (or other transistor parameters) will degrade significantly, and the overall system will lose precision in its operation.

This Appendix describes an $O(M)$ complexity circuit technique (which can be represented by the topology in Fig. 3.1(b)) for implementing either WTA and/or MAX circuits, based on current-mode principles. The resulting circuit does not rely on the matching of an M-size transistor array. The precision of the overall system relies on precise current replication, which can be achieved locally without matching M transistors. Sometimes, when assembling large neural and/or fuzzy systems, a WTA/MAX circuit must be distributed among several chips [3.13]. The circuit described here can be distributed among several chips with no influence on its precision, as shown in the Section on experimental results.

In Section 3.2 a mathematical model that performs WTA/MAX operation is described. This operation principle will be used in Section 3.3 to develop a current-mode processing circuit. Sections 3.4 and 3.5 deal with stability considerations of the circuit presented in Section 3.3. Finally, Section 3.6 provides experimental measurement results obtained from prototypes fabricated in two different CMOS technologies, and from multi-chip systems formed by chips of the same or different technologies.

3.2. Operation Principle

The operation principle given in this Section can be used for simultaneous implementation of a WTA and MAX circuit. The system has M cells. Each cell j produces an output

$$I_{oj} = \alpha_j H(T_j - I_o) \quad , \quad j = 1, \ldots M \tag{3.1}$$

where

2. If the inhibition that goes from cell i to cell j does not depend on j.

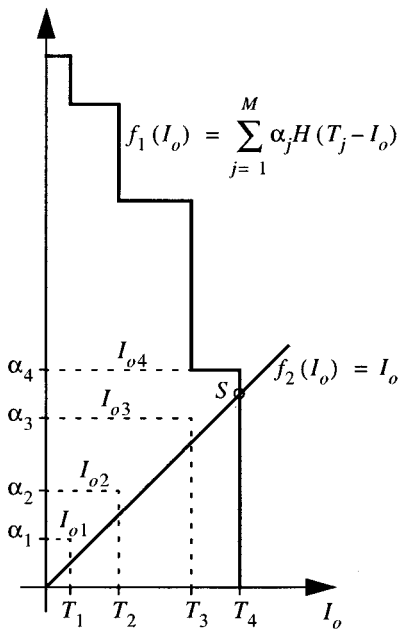


Fig. 3.2: Graphic Representation of the Solution of eq. (3.4)

$$I_o = \sum_{j=1}^{M} I_{oj}, \tag{3.2}$$

$H(\cdot)$ is the step function defined as

$$H(x) = \begin{cases} 1 & , \ x \geq 0 \\ 0 & , \ x < 0 \end{cases} \tag{3.3}$$

and T_j is the external input to the j-th cell. Substituting eqs. (3.1) into eq. (3.2) yields

$$I_o = \sum_{j=1}^{M} \alpha_j H(T_j - I_o) \ . \tag{3.4}$$

Fig. 3.2 shows a graphic representation of the functions $f_1(I_o) = \Sigma_j \alpha_j H(T_j - I_o)$ and $f_2(I_o) = I_o$. The intersection of $f_1(I_o)$ and $f_2(I_o)$ provides the solution to eq. (3.4). Note that if $\alpha_j > 0 \quad \forall j$, eq. (3.4) has a unique equilibrium point S, as deduced from Fig. 3.2. Furthermore, if

$$\alpha_j \geq T_j \quad , \quad \forall j \tag{3.5}$$

the value of I_o at the equilibrium point S is

$$I_o\big|_S = max\{T_j\} \tag{3.6}$$

and the cell that drives a nonzero output $I_{oj} \neq 0$ is the winner. Consequently, a circuit that implements eq. (3.4) can be used to realize both a WTA or a MAX circuit.

In the case of an LTA or a MIN circuit, the same mathematical model of Fig. 3.2 applies if each input equals

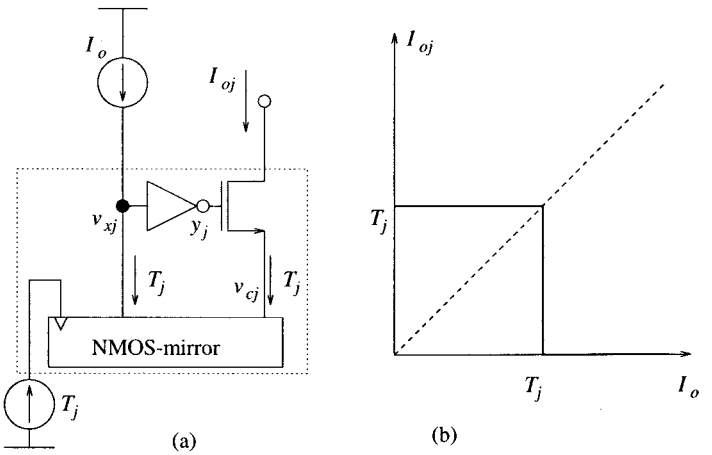


Fig. 3.3: WTA unit cell: (a) circuit diagram, (b) transfer curve

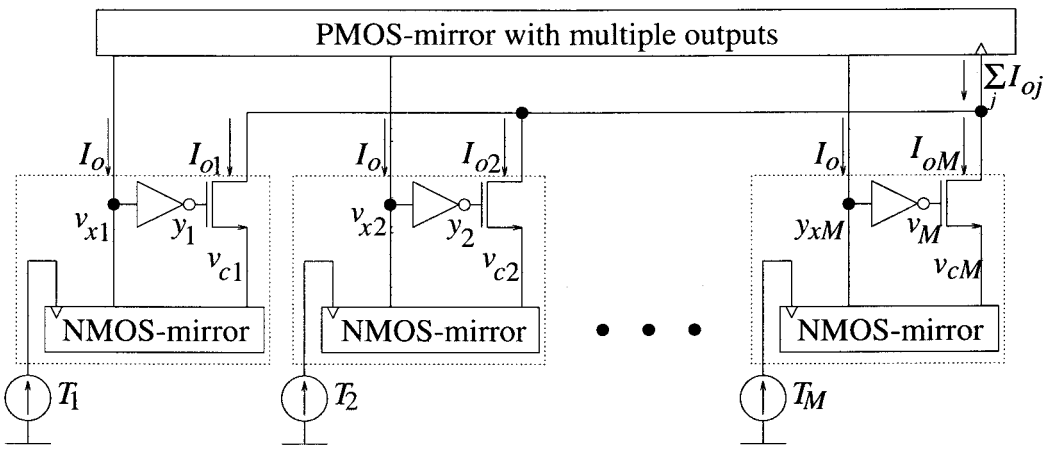


Fig. 3.4: Diagram of the WTA circuit

$$I_L - T_j, \tag{3.7}$$

where I_L is an upper bound for all input

$$0 \le T_j \le I_L \quad , \quad \forall j. \tag{3.8}$$

3.3. Circuit Implementation

This Section shows how to realize a circuit that implements eq. (3.4) using currents to represent the mathematical variables T_j and I_o. The circuit for each cell j is shown in Fig. 3.3. It consists of a 2-output current mirror, a MOS transistor, and a digital inverter. Each cell j receives two input currents, T_j and I_o, and delivers one output current I_{oj}. The inverter acts as a current comparator. If $I_o > T_j$ the inverter output y_j is low, the MOS transistor is OFF, and I_{oj} is zero. If $I_o < T_j$ the inverter output y_j is high, the MOS transistor is ON, and $I_{oj} = T_j$. Consequently, the circuit of Fig. 3.3 implements a cell with $\alpha_j \equiv T_j$.

Fig. 3.4 shows the complete WTA or MAX circuit. It consists of M cells shown in Fig. 3.3 and an additional M-output current mirror. Note that the responsibility of the M-output current mirror is to deliver the sum of currents $I_o = \Sigma_j I_{oj}$ to each of the M cells. Replication and transportation of current I_o must be very precise. If the number of cells M is too large, or if the circuit has to be distributed among several chips, high

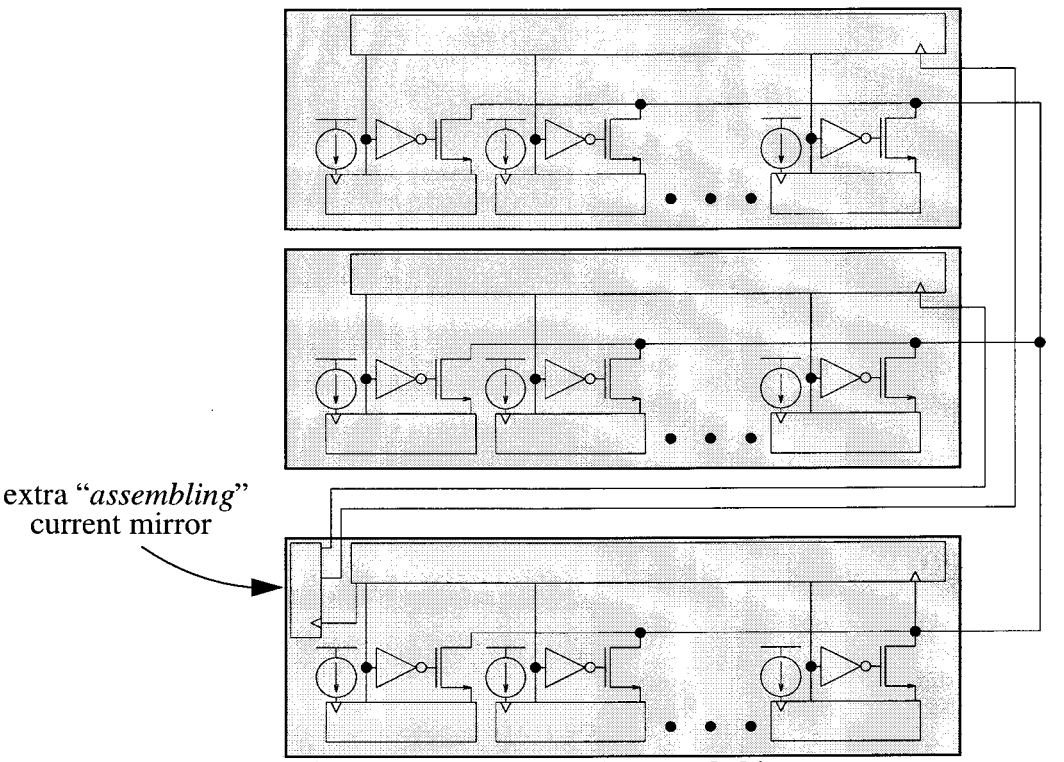


Fig. 3.5: Strategy to assemble several chips

precision in I_o replication cannot be guaranteed by a single current mirror with M outputs. In this case, replication of current I_o must rely on several mirrors with a smaller number of outputs but with guaranteed precise replication. Fig. 3.5 shows an arrangement to distribute the circuit of Fig. 3.4 among several chips. The fact that current I_o can be replicated many times without relying on the matching of a large array of transistors is the advantage of this WTA and MAX (or LTA and MIN) circuit technique over other implementations.

The precision of the overall WTA current mode circuit is determined by the kind of current mirrors used. Since the current comparator has virtually no offset, the current error at the input of each current comparator is determined by mirror mismatches. The error at the positive current I_o available at the input of each current comparator results from one p-mirror reflection, preceded by an n-mirror reflection of the winning cell,

$$\sigma^2(I_o) = \sigma_N^2 + \sigma_P^2 \tag{3.9}$$

while the error of the negative current T_j at the current comparator inputs results from a single n-mirror reflection,

$$\sigma^2(T_j) = \sigma_N^2. \tag{3.10}$$

The total current error at the input of each current comparator is therefore given by,

$$\sigma_{Total}^2 = \sigma^2(I_o) + \sigma^2(T_j) = 2\sigma_N^2 + \sigma_P^2 \tag{3.11}$$

However, the error introduced by a current mirror is not only the random mismatch contribution, as considered in eqs. (3.9)-(3.11), but also its systematic error contribution, which results from different drain-to-source

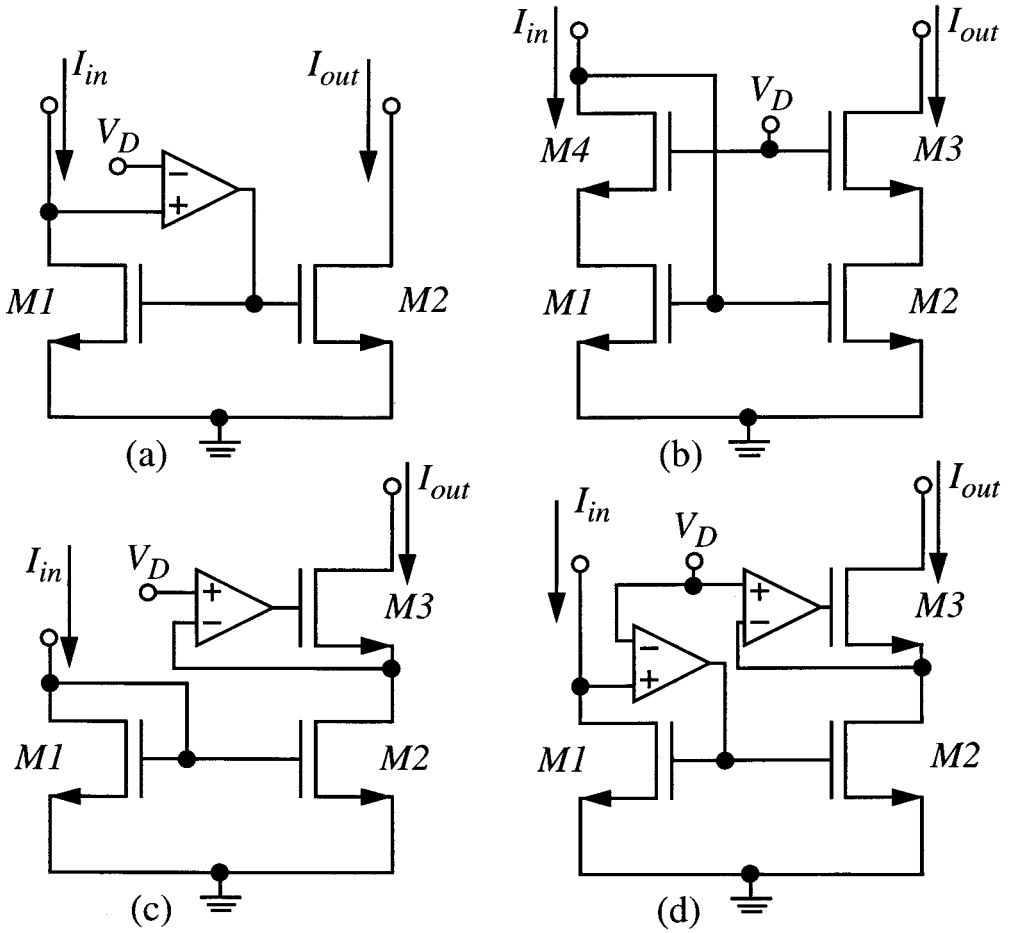


Fig. 3.6: Enhanced current mirror topologies. (a) active, (b) cascode, (c) regulated cascode output, and (d) active regulated cascode

voltages at the reflecting transistors, poor impedance coupling, and inherent nonlinear MOS transistor operation. In our implementation we have chosen an "active-input regulated-cascode" current mirror (see Fig. 3.6(d)) [3.2]. As shown in Fig. 3.6(d), this topology maintains the same drain-to-source voltage at the input and output reflecting transistors, thus avoiding this source of systematic error component.

The active current mirror idea (see Fig. 3.6(a)) [3.14] was introduced as a need to maintain a constant voltage at the input of a current mirror in order to avoid current subtraction errors in previous stages. This technique allows the V_{GS} voltage of the current mirror input transistor *M1* to be independent of its V_{DS} voltage, which will be kept constant, and therefore lowering the mirror input impedance and minimizing loading effects on previous stages. The current mirror will be operative as long as transistors *M1* and *M2* are kept in saturation. The higher the reference voltage V_D is, more current is allowed through the mirror with *M1* operating in saturation. The drain-to-source voltage of the output transistor depends on the load of the mirror and will be dependent on the mirror current. If the load impedance is not sufficiently low *M2* will suffer of large drain-to-source voltage variations, which through the channel length modulation effect, will cause a systematic mismatch error between the input and output currents of the mirror. Such a circumstance can be avoided by using the cascode current mirror of Fig. 3.6(b). However, this mirror has a smaller output voltage swing and requires a high input voltage drop for high currents (which are needed for maximum accuracy) [3.16]. The regulated-cascode output stage (see Fig. 3.6(c)[3]) [3.17] would allow to maintain the V_{DS} of

transistor *M2* constant and to increase significantly the output impedance of the mirror, while not sacrificing voltage range at the input of the current mirror. This current mirror will be operative as long as transistor *M2* remains in saturation, which depends on V_D and the input current, as well as on the load impedance at the output of the mirror.

By combining the active current mirror input of Fig. 3.6(a) with the regulated-cascode output in Fig. 3.6(c), the *active-input regulated-cascode current mirror* of Fig. 3.6(d) results. This current mirror has a very low input impedance, a very high output impedance and is operative if transistors *M1* and *M2* are either in saturation or ohmic region (because their V_{DS} voltages are always equal). Therefore, the gate voltage of transistors *M1* and *M2* can change from rail to rail. However, this current mirror will fail to operate with high accuracy if the output voltage approaches V_D. But, on the other hand, V_D can be made smaller than for Fig. 1(a) because *M1* and *M2* can operate now in ohmic regime.

When cascading current mirrors, the regulated-cascode output is not needed and the configuration of Fig. 3.6(a) can be used. The virtual ground effect at the drain of transistor *M2* is produced by the active input of the next current mirror. However, in this case V_D has to be set equal for all PMOS and NMOS active current mirrors. Also, care has to be taken by choosing the value of V_D in order to respect the output voltage range of the circuit at the input of the first mirror, and to respect the input voltage range of the circuit at the output of the last current mirror.

As it has been previously explained, the accuracy limitation of current mirrors has two main types of sources, systematic errors and random errors. Systematic errors are caused by different V_{DS} voltages at transistors *M1* and *M2* due to poor input/output impedance coupling between subsequent stages and high-order nonlinear effects. Random errors are fundamentally caused by differences in the electrical parameters between transistors *M1* and *M2*, due to random process parameter variations. While random mismatch error contributions are practically independent on the circuit topology, systematic errors change considerably from one topology to another. We will consider that the total precision of a current mirror is given by

$$\Delta I_{Total} = \Delta I_{sys} + \sigma_I \tag{3.12}$$

where ΔI_{sys} is the systematic error contribution (evaluated through a single nominal Hspice simulation) and σ_I is the standard deviation of the output current (evaluated through 30 Hspice Monte Carlo simulations). The statistical significance of σ_I is that 68% of the samples have an output current error within the range $(\Delta I_{sys} - \sigma_I, \Delta I_{sys} + \sigma_I)$. For random mismatch errors Hspice simulations it is considered that the only sources of random mismatch are the differences in threshold voltage (V_T) and current factor ($\beta = C_{ox}\mu W/L$), and that their standard deviation is given by [3.16],

3. Although we are showing a differential input voltage amplifier, in the original paper [3.17] a single input amplifier was used. In this case the voltage V_D would be set through process and circuit parameters.

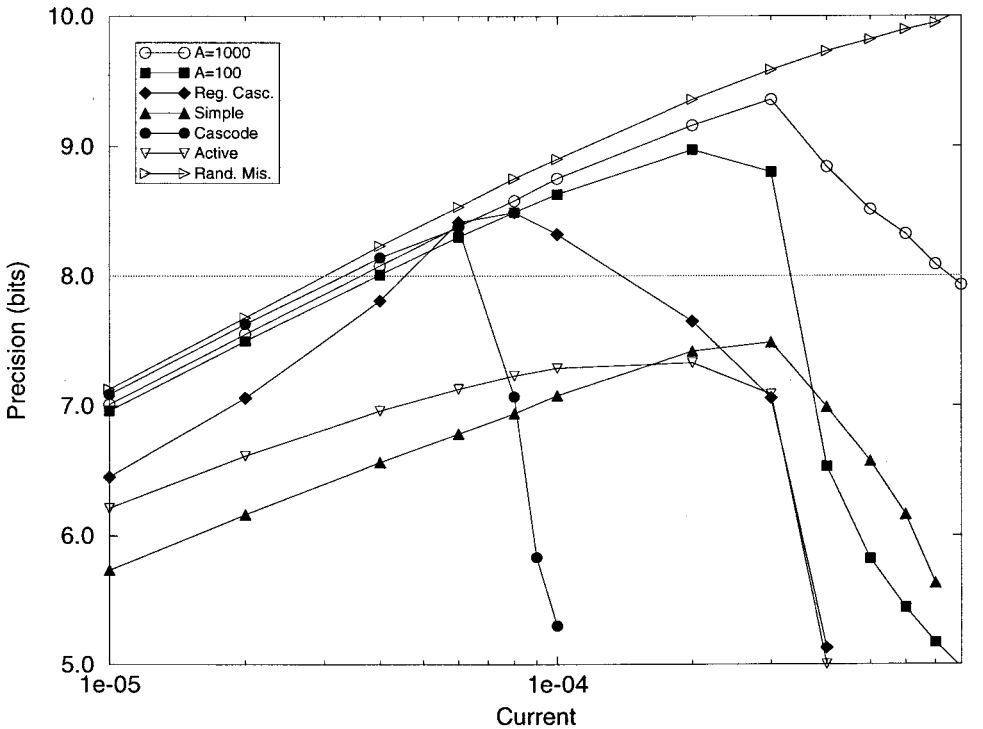


Fig. 3.7: Resolution (in bits) as a function of working current for different current mirror topologies

$$\sigma^2(V_T) = \frac{A_{V_T}^2}{WL} + S_{V_T}^2 D^2$$
$$\frac{\sigma^2(\beta)}{\beta^2} = \frac{A_\beta^2}{WL} + S_\beta^2 D^2 \qquad (3.13)$$

where W and L are the sizes of the transistors, D (~W) their separation, and A_{VT}=15$mV\mu m$, S_{VT}=2$\mu V/\mu m$, A_β=2.3%μm, S_β=2×10$^{-6}\mu m$ (parameters given in [3.16] for a 1.6μm N-well process with 25nm gate oxide and direct wafer writing).

In the following simulations we use W=100μm and L=20μm for transistors *M1* and *M2* of all current mirror topologies, with V_D=1.5V (power supply is 5V). Fig. 3.7 represents the total accuracy of eq. (3.12) as a function of operating current for different current mirror topologies. The first two topologies are for the *active-input regulated-cascode current mirror* with amplifier gain values *A=1000* and *A=100*. For *A=1000* the resolution is above 8-bits for current values between *40*μA and *750*μA, while for *A=100* it is only between *40*μA and *330*μA. In both cases the decrease in precision above ~*300*μA is because transistors *M1* and *M2* enter their ohmic region of operation. A simple current mirror has a resolution below 8-bits for the complete current range. This is mainly due to poor impedance coupling, which can be avoided with regulated cascode outputs, achieving 8-bits resolution between *45*μA and *150*μA. The cascode mirror suffers from large voltage drops at its input, thus offering the 8-bits resolution only between *40*μA and *70*μA. The active-input mirror (Fig. 3.6(a)) has low resolution because its output voltage was set to 2.5V, hence rendering an important systematic error contribution. Also shown in Fig. 3.7 is the random mismatch error contribution (σ_I in eq. (3.12)), which is approximately the same for all the topologies. Note that all topologies suffer from loss of

precision at high currents. This is produced because the increasing voltage drops at the different nodes approximate the limit of available voltage range.

The use of the active input requires stability compensation [3.14]. Compensation of an active-input current mirror for an operating current of several decades and achieving speed figures better than for standard current mirror topologies is not a simple task. For our application, where we needed to achieve 8-bits accuracy for a one-decade operating current and with speeds faster than *100ns* (for most part of the range), we required the use of an automatic transistor sizing optimization tool [3.15] for the design of the voltage amplifiers. For our application we required voltage amplifiers with a gain lower than *1000*, therefore a simple topology could be used ([3.14] or [3.18]). The regulated cascode output amplifier does not require compensation.

Table 3.1 shows the transistor level simulated transient times of our design when using a regular OTA [3.18] as the active device. These transients correspond to the time the mirror takes to settle to 1% of the final current value, when the input is driven by an ideal current step signal changing between the levels given in the first row of the table. Table 3.1 also shows the delays for other topologies that have the same transistor sizes. Note that speed is improved with respect to mirrors that do not use active devices. The reason is that the active device introduces more design variables, thus allowing a more optimum final result.

	10μA to 15μA	50μA to 75μA	100μA to 150μA	250μA to 375μA	400μA to 600μA	530μA to 800μA
act. reg. casc.	65ns	70ns	45ns	45ns	70ns	460ns
simple	160ns	80ns	65ns	40ns	80ns	35ns
cascode	180ns	85ns	-	-	-	-
active input	65ns	70ns	45ns	35ns	45ns	-
reg. cas. out.	165ns	80ns	65ns	50ns	-	-

Table 3.1. Simulated Transient Times

As we will see in Section 3.6, we built a CMOS prototype of the circuit in Fig. 3.4 (and Fig. 3.5) using simple current mirrors for the NMOS mirrors, and active-input mirrors for the PMOS and the assembling current mirrors. As it will be discussed in Section 3.6, this is sufficient to guarantee multichip WTA operation. For precise MAX operation, however, more elaborated mirrors would be needed for the NMOS mirrors.

3.4. System Stability Coarse Analysis

Let us assume that the dynamics of each cell (see Fig. 3.3) can be modelled by the following first-order nonlinear differential equation

$$C_c \dot{v}_{xj}(t) + G_c (v_{xj}(t) - v_M) + T_j = I_o(t) \tag{3.14}$$

where C_c is the total capacitance available at node v_{xj}, G_c is the total conductance at this node, and v_M is the inverter trip voltage. Let us also assume that the output current of a cell is given by

$$I_{oj}(t) = T_j U(v_M - v_{xj}(t)) \tag{3.15}$$

where $U(\cdot)$ is a continuous and differentiable approximation to the step function of eq. (3.3). For example, we can define $U(\cdot)$ as the following sigmoidal function,

$$U(x) = \frac{1}{1+e^{-x/\varepsilon}} \tag{3.16}$$

where ε is positive and non-zero but close to zero. Now consider eq. (3.14) for two nodes, *j* and *w*. Let *w* be the node that eventually should become the winner. If we subtract eq. (3.14) for the two nodes *j* and *w*, then

$$C_c[\dot{v}_{xj}(t) - \dot{v}_{xw}(t)] + G_c[v_{xj}(t) - v_{xw}(t)] = T_w - T_j \tag{3.17}$$

Eq. (3.17) has the following solution

$$v_{xj}(t) - v_{xw}(t) = \frac{T_j - T_w}{G_c} + \left[v_{xj}(0) - v_{xw}(0) - \frac{T_j - T_w}{G_c}\right] e^{-\frac{t}{\tau_c}} \quad , \quad \tau_c = \frac{C_c}{G_c}. \tag{3.18}$$

After a few time constants τ_c the difference between the two node voltages will remain constant and equal to their difference at the equilibrium point. Therefore, if we can obtain the expression for $v_{xw}(t)$, applying eq. (3.18) would obtain $v_{xj}(t)$ for the rest of the nodes.

Consider now eq. (3.14) for node *w*, and substitute eqs. (3.2) and (3.15) into it,

$$C_c\dot{v}_{xw}(t) + G_c(v_{xw}(t) - v_M) + T_w = \sum_j T_j U(v_M - v_{xj}(t)) . \tag{3.19}$$

Since $v_{xj}(t)$ is given by eq. (3.18), after a few time constants τ_c eq. (3.19) becomes

$$C_c\dot{v}_{xw}(t) = G_c(v_M - v_{xw}(t)) - T_w + \sum_j T_j U\left(v_M - v_{xw}(t) + \frac{T_w - T_j}{G_c}\right). \tag{3.20}$$

This first order differential equation has stable equilibrium points if

$$\left.\frac{d\dot{v}_{xw}}{dv_{xw}}\right|_{equilibrium\ point} < 0. \tag{3.21}$$

By deriving eq. (3.20) with respect to v_{xw} results

$$C_c\frac{d\dot{v}_{xw}}{dv_{xw}} = -G_c - \sum_j T_j U'(\cdot) . \tag{3.22}$$

Since G_c, I_j, and $U'(\cdot)$ are always positive, eq. (3.22) is always negative for all possible values of v_{xw} (including the equilibrium point). Consequently, eq. (3.20) represents the dynamics of a stable system.

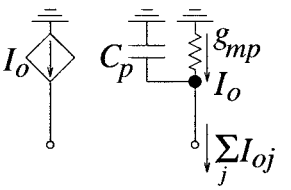


Fig. 3.8: Small Signal Modelling of Delay of the *N*-output Current Mirror

The discussion in this Section assumes that the *M*-output current mirror presents no delay. This is not very realistic. If we assume the *M*-output current mirror of Fig. 3.4 has the first-order dynamics defined by the small signal equivalent circuit depicted in Fig. 3.8.

Current $I_o(t)$ represents each of the M outputs of this current mirror, and $\Sigma_j I_{oj}$ its input. The dynamics of this current mirror in time-domain are given by

$$I_o(t) + \tau_p \dot{I}_o(t) = \sum_j I_{oj}(t) \quad , \quad \tau_p = \frac{C_p}{g_{mp}}. \tag{3.23}$$

Eqs. (3.14)-(3.16) are still valid, but eqs. (3.17)-(3.19) have a higher order dynamics. By substituting eq. (3.14) and its derivatives into eq. (3.23) results

$$\tau_p C_c \ddot{v}_{xw}(t) + (C_c + \tau_p G_c)\dot{v}_{xw}(t) + G_c v_{xw}(t) = (G_c v_M - T_w) + \sum_{j=1}^{M} T_j U(v_M - v_{xj}(t)) . \tag{3.24}$$

Subtracting them for two nodes j and w yields,

$$\tau_p C_c [\ddot{v}_{xj}(t) - \ddot{v}_{xw}(t)] + (C_c + \tau_p G_c)[\dot{v}_{xj}(t) - \dot{v}_{xw}(t)] + G_c [v_{xj}(t) - v_{xw}(t)] = T_w - T_j . \tag{3.25}$$

The solution to this differential equation is

$$v_{xj}(t) - v_{xw}(t) = \frac{T_w - T_j}{G_c} + K_1 e^{-\frac{t}{\tau_p}} + K_2 e^{-\frac{t}{\tau_c}} \quad , \quad \tau_c = \frac{C_c}{G_c} \tag{3.26}$$

where K_1 and K_2 are determined by initial conditions. Consequently, after a few time constants τ_c and τ_p, eq. (3.24) would be given by

$$\tau_p C_c \ddot{v}_{xw}(t) + (C_c + \tau_p G_c)\dot{v}_{xw}(t) + G_c v_{xw}(t) = (G_c v_M - T_w) + \sum_{j=1}^{M} T_j U\left(v_M - v_{xw}(t) + \frac{T_w - T_j}{G_c} \right) \tag{3.27}$$

If $v_{xw}(t)$ is well above or below $v_M + (T_w - T_j)/G_c$, then the corresponding j-th cell function $T_j U(\cdot)$ will equal either 0 or T_j, respectively. In these cases the term $T_j U(\cdot)$ contributes to the constant term (time independent) of eq. (3.27). On the other hand, if $v_{xw}(t)$ is close to $v_M + (T_w - T_j)/G_c$, the term $T_j U(\cdot)$ is close-to-linearly dependent on $v_{xw}(t)$. In this case, its first-order Taylor series expansion is

$$T_j U(\cdot) \approx \frac{1}{2} T_j + T_j U'(0)\left(v_M + \frac{T_w - T_j}{G_c} - v_{xw}(t) \right). \tag{3.28}$$

This term contributes to the constant term and to the $v_{xw}(t)$ term of eq. (3.27). Summing over all cells obtains

$$\sum_{j=1}^{M} T_j U(\cdot) \approx -Sv_{xw}(t) + K, \tag{3.29}$$

where K and S are constants and $S > 0$ (because I_j, $U'(0) > 0$). Therefore, the poles of eq. (3.27) are the roots of

$$\tau_p C_c s^2 + (C_c + \tau_p G_c)\, s + (G_c + S) = 0 \tag{3.30}$$

which always have a negative real part. Consequently, eq. (3.27) converges always to its unique equilibrium point.

3.5. System Stability Fine Analysis

Performing electrical simulations of the circuit in Section 3.3, verifies that the analysis in Section 3.4 is a good approximation as long as the equilibrium point does not lie in the transition region of any of the M sigmoidal functions $U(\cdot)$. This can only be guaranteed if $\alpha_j = T_j$ and the two largest inputs T_j and T_w are sufficiently different. If $\alpha_j > T_j$ or (with $\alpha_j = T_j$) if two or more inputs T_j are maximum and very similar, the equilibrium point of the system (see Fig. 3.2) will be in the transition region of some sigmoids $U(\cdot)$. In these cases, transistor parasitic elements that have been neglected in the analysis of Section 3.4 may render unstable behavior. Consequently, some kind of compensation is necessary.

Under unstable conditions the system exhibits the following characteristics (observed through electrical simulations with Hspice):

- Only the cells j whose sigmoid functions $U(\cdot)$ must be in their transition region at the equilibrium point are unstable. The rest of the cells behave as if the system had reached its equilibrium point.
- The unstable cells present oscillations (presence of complex conjugate poles).
- In the case of $\alpha_j = T_j$ and with two or more equal maximum inputs, the steady-state oscillating waveforms at these cells become the same, regardless of their initial conditions.

This last observation suggests that a stability analysis could be performed by simply considering one cell in the system, which represents the parallel connection of all unstable cells, as shown in Fig. 3.9(a). On the other hand, since the unstable cells have the equilibrium point in the transition region of their sigmoid $U(\cdot)$, we can linearize these sigmoids for the stability analysis. Therefore, let us consider the small signal equivalent circuit shown in Fig. 3.9(b), where the circuitry comprised by dashen lines represents the parallel of all cells with equal and maximum input. The rest of the circuitry models the M-output current mirror (or set of current mirrors) responsible for distributing the global current I_o among the M cells. The minimum set of dynamic elements needed for the system to present unstable oscillating behavior are parasitic capacitors C_c, C_p, and C_g (observed through electrical simulation).

The frequency-domain KCL equations of the linear circuit of Fig. 3.9(b) are

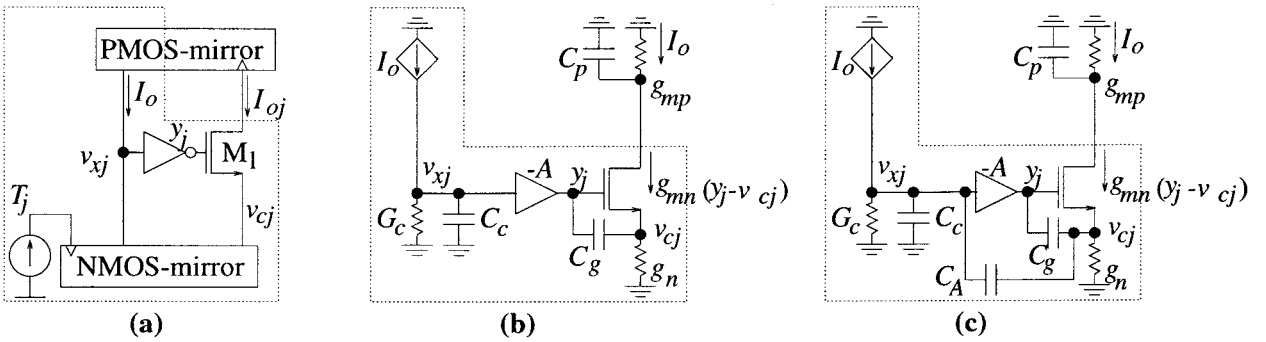


Fig. 3.9: (a) Parallel connection of unstable cells, (b) uncompensated small signal equivalent circuit, (c) compensated small signal equivalent circuit.

$$\begin{aligned} I_o &= V_{xj}(G_c + sC_c) \\ g_{mn}(y_j - V_{cj}) &= g_n V_{cj} - sC_g(y_j - V_{cj}) \\ M_m g_{mn}(y_j - V_{cj}) &= \left(1 + s\frac{C_p}{g_{mp}}\right) I_o \\ y_j &= -AV_{xj} \end{aligned} \tag{3.31}$$

Routine analysis yields the following third-order polynomial

$$\begin{aligned} as^3 + bs^2 + cs + d &= 0 \\ a &= \frac{C_p C_c C_g}{g_{mp}} \\ b &= C_p C_g \frac{G_c}{g_{mp}} + C_p C_c \frac{g_{mn}}{g_{mp}} + C_g C_c \\ c &= C_g G_c + C_p G_c \frac{g_{mn}}{g_{mp}} + C_c g_{mn} \\ d &= g_{mn} G_c + A M_m g_n g_{mn} \end{aligned} \tag{3.32}$$

Since all coefficients a, b, c, and d are positive, the roots of this polynomial have negative real parts if $bc - ad > 0$. Considering that parasitic capacitances C_p, C_c, and C_g are approximately of the same order of magnitude and that $g_{mp} > g_{mn} \gg g_n \approx G_c$, the stability condition simplifies to

$$AM_m < \frac{C_c}{g_n}\left(\frac{g_p}{C_p} + \frac{g_{mn}}{C_g}\right) \tag{3.33}$$

where M_m is the number of cells with equal and maximum input. This condition is not easy to satisfy since A must be large for proper operation, M_m may become large, and it is not trivial to make the right hand side of eq. (3.33) very large.

Stability compensation can be achieved by introducing capacitor C_A, as shown in Fig. 3.9(c).

The frequency domain KCL equations of the linear circuit of Fig. 3.9(c) are

$$\begin{aligned} I_o &= V_{xj}(G_c + sC_c) + sC_A(V_{xj} - V_{cj}) \\ g_{mn}(y_j - V_{cj}) &= g_n V_{cj} - sC_g(y_j - V_{cj}) - sC_A(V_{xj} - V_{cj}) \\ M_m g_{mn}(y_j - V_{cj}) &= \left(1 + s\frac{C_p}{g_{mp}}\right) I_o \\ y_j &= -AV_{xj} \end{aligned} \tag{3.34}$$

Routine analysis yields the following third-order polynomial

$$\begin{aligned} as^3 + bs^2 + cs + d &= 0 \\ a &= \tau_p [C_c C_g + C_c C_A + (A+1) C_g C_A] \\ b &= \tau_p C_c (g_{mn} + g_n) + \tau_p C_A g_n + \tau_p C_g G_c + C_g C_c + \tau_p G_c C_A + C_c C_A + \tau_p (A+1) g_{mn} C_A + (A+1) C_A C_g \\ c &= (\tau_p G_c + C_c)(g_{mn} + g_n) + C_A (g_{mn} + g_n + AM_m g_{mn} + Ag_{mn} + M_m g_{mn}) \\ d &= AM_m g_{mn} g_n \end{aligned} \tag{3.35}$$

Assuming $A \gg 1$, $g_{mp} > g_{mn} \gg g_n \approx G_c$ eqs. (3.35) can be simplified to

$$\begin{aligned} a &\approx A\frac{C_g C_A C_p}{g_{mp}} \\ b &\approx AC_A\left(\frac{g_{mn}}{g_{mp}} C_p + C_g\right) \\ c &\approx C_A (AM_m + A + M_m) g_{mn} \\ d &\approx AM_m g_{mn} g_n \end{aligned} \tag{3.36}$$

Since all coefficients *a, b, c,* and *d* are positive, the roots of this polynomial have negative real parts if $bc - ad > 0$, which yields the following stability condition,

$$\left(1 + \frac{1}{M_m}\right) C_A > \frac{g_n}{g_{mp}/C_p + g_{mn}/C_g}. \tag{3.37}$$

The worst case occurs for very large values of M_m, for which eq. (3.37) reduces to

$$C_A > \frac{g_n}{g_{mp}/C_p + g_{mn}/C_g}. \tag{3.38}$$

Note that now the stability condition does not depend on gain A, and is easier to fulfill. However, now capacitor C_A degrades the settling speed of the system. Capacitor C_A acts as a Miller capacitance. Since the DC-gain from node v_{xj} to node v_{cj} is approximately $-A$ (i.e. the negative of the slope of $U(\cdot)$), there will be an effective Miller capacitance of value $(A+1)C_A$ in parallel with the original C_c capacitor. If the sigmoid is not in its transition region $A \approx 0$, but if the sigmoid is in its transition region A can be very large. Therefore, for compensated cells eq. (3.20) must be changed to

$$[C_c + C_A + U'(v_M - v_{xw}) C_A] \dot{v}_{xw} = G_c (v_M - v_{xw}) - T_w + \sum_j T_j U\left(v_M - v_{xw} + \frac{T_w - T_j}{G_c}\right). \tag{3.39}$$

If the winning cell is in its transition region $U'(v_M - v_{xw}) \neq 0$ and a large capacitance $C_c + (A+1)C_A$ is present at node v_{xw}. Otherwise, $U'(v_M - v_{xw}) = 0$ and the effective capacitance is only $C_c + C_A$.

3.6. Experimental Results

A WTA-MAX system with M=10 competing cells has been designed and fabricated in two different technologies. The first prototype has been integrated in a double-metal single-poly 1.0μm CMOS technology (ES2), and the other in a double-metal double-poly 2.5μm CMOS process (MIETEC). Both technologies were available through the European silicon foundry service, EUROCHIP.

If the circuit is going to be used as a MAX circuit, all current mirrors must provide good replication precision. They need to have small systematic errors and small random deviations [3.16], so that the resulting value of current I_o resembles the maximum among all inputs as much as possible. However, if the circuit is going to be used as a WTA circuit, requirements are not that severe. If inside one single chip, a WTA performs the same even if the current mirrors have appreciable systematic errors. Since systematic errors are common with respect to all inputs, the system can still determine which input is maximum. On the other hand, random mismatch errors in the current mirrors must be kept small because these errors change randomly from one input to another. Reducing random errors implies using larger transistor sizes. Reducing systematic errors implies using more elaborate current mirror topologies that either reduce their output conductance (using cascode [3.16], regulated cascode [3.17], or gain-boosting [3.20] techniques), decrease their input impedance [3.21], or both [3.2].

For our application it was not critical that the final value of I_o be an exact replica of the maximum of the input. Therefore, we used a simple 3-transistor current mirror (without any output conductance or input impedance decreasing technique) for the 2-output NMOS current mirror of each cell. However, we used active input current mirrors [3.21] for the M-output PMOS current mirror and for the extra NMOS assembling current mirror (see Fig. 3.5). These current mirrors assure fixed voltages at their input nodes. This was necessary because if the system is distributed among several chips, the presence of the assembling current mirror would break the symmetry between some of the inputs, making systematic errors affect these inputs differently.

The following presents proper system operation of a WTA circuit in one single chip, in two chips of the same technology, and in two chips each of a different technology. As will be shown, the DC-behavior of the system is not degraded when the operation is distributed among several chips. In the remainder of this Section we will detail experimental measurements related to the precision of a WTA and its speed response.

A. Operation Precision

The DC transfer curves of the system have been measured for different input current levels and for different system configurations. Fig. 3.10 shows thirty transfer curves when the competing cells are inside the same chip. Each curve is obtained by randomly selecting a pair of input cells, i and j, applying a constant input current $T_i = T_P$ to the first, and sweeping the input current of the second T_j from $0.9 \times T_P$ to $1.1 \times T_P$. The

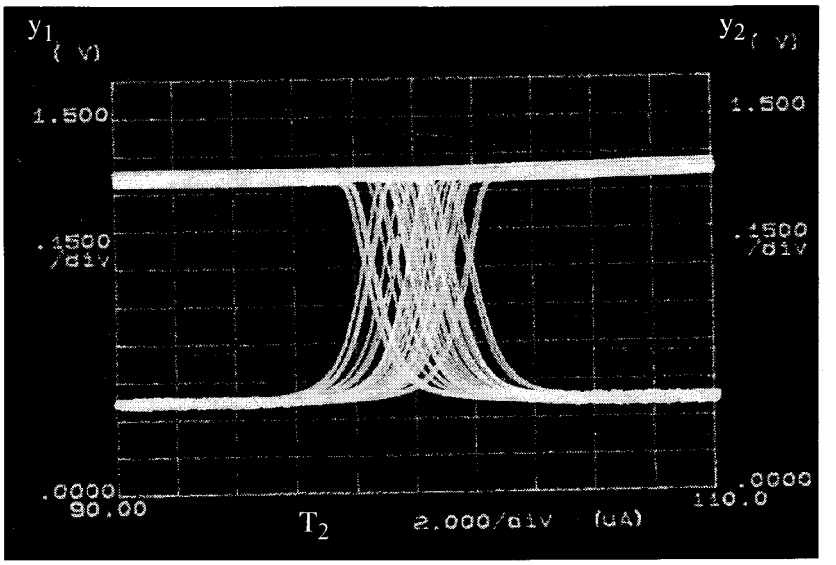


Fig. 3.10: Transfer curves of the WTA implemented in a ES2 1.0μm chip for an input current level of 100μA

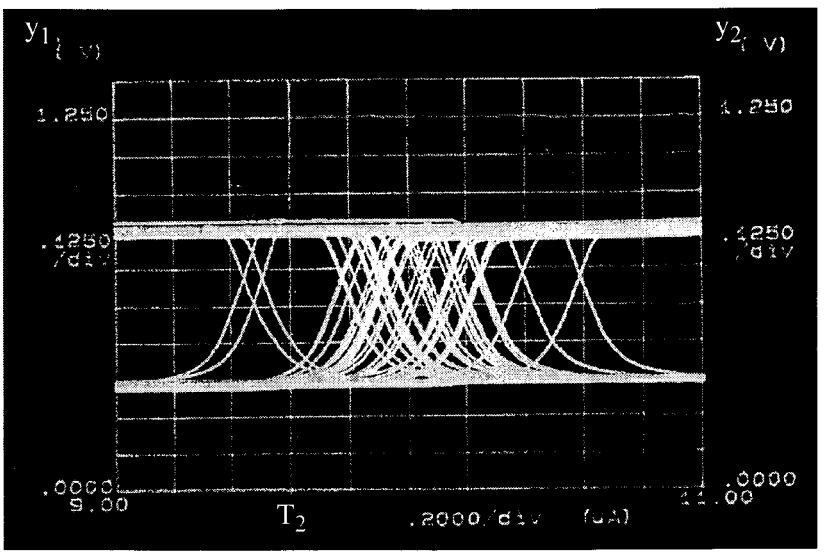


Fig. 3.11: Transfer curves when two ES2 1.0μm chips are assembled and for an input current level of 10μA

figure represents the two inverter output voltages, y_i and y_j, versus the current T_j. For each pair of cells, i and j, we measure the value of T_j at the point where $y_i = y_j$. Let us call this value T_M. Thirty curves were measured for each value of T_P, resulting in thirty values of T_M. The difference between the mean of these thirty T_M values and T_P is a measure of the systematic error of T_M. Let us call it $\varepsilon(T_P)$. The variance of the thirty T_M values represents the random error of T_M. Let us call it $\sigma(T_P)$. In the case of Fig. 3.10, corresponding to a WTA inside one single chip fabricated in the ES2 1.0μm CMOS technology with $T_P = 100\mu A$, we measured a random deviation of $\sigma(T_P) = 1.04\%$ and a systematic error of $\varepsilon(T_P) = 0.03\%$.

Fig. 3.11 again shows thirty DC transfer curves, where $T_P = 10\mu A$ and the system is built by assembling two chips of the same technology using the set-up illustrated in Fig. 3.5. To obtain these curves, cell i was always chosen among the cells in the first chip, and cell j was always selected from the second chip. A random

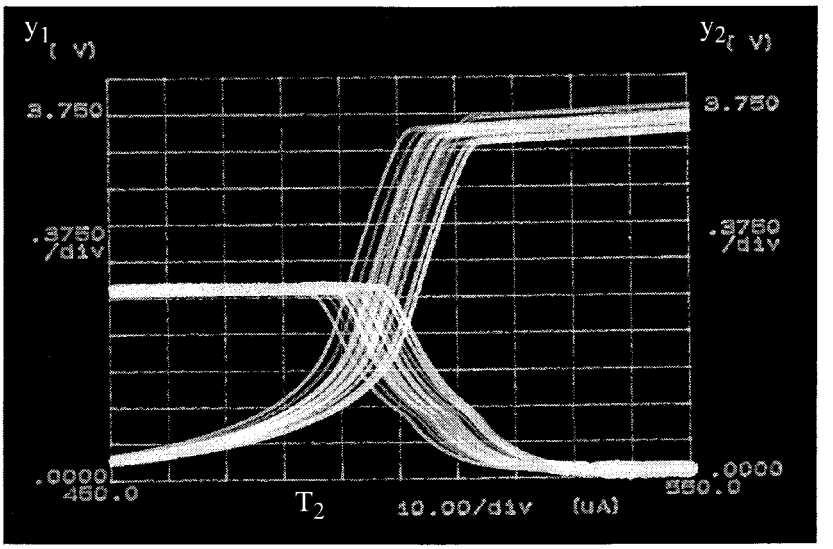


Fig. 3.12: Transfer curves when two chips of different technology are assembled and for an input current level of 500μA

deviation of $\sigma(T_P) = 2.30\%$ was measured, while the systematic deviation $\varepsilon(T_P) = 0.05\%$. Fig. 3.11 shows thirty DC transfer curves for the case $T_P = 500\mu A$, when two chips of different technologies are used. In this case $\sigma(T_P) = 2.18\%$ and $\varepsilon(T_P) = 0.06\%$. Note that the voltage ranges of y_i and y_j differ for the two chips.

Table 3.3 contains the measured total error (defined as $\sigma(T_P) + \varepsilon(T_P)$) for three decades of change in T_P. The table shows results for the cases of WTAs inside one chip, assembled using two chips of the same technology, and assembled with two chips of different technologies. Note that the precision degradation is very small when the system is distributed among two chips, regardless of whether the chips are of the same technology or not. This is the main advantage of this WTA-MAX circuit with respect to others reported in literature. This is shown in Table 3.3 which depicts the simulation results of another WTA [3.12]. The input signals for this WTA are voltages that range from 1.5V to 4.5V. As can be seen, there is a significant precision degradation when the WTA is distributed among two chips of different technologies caused by a large increase in the systematic error component [3.22].

B. Operation Speed

Delay measurements were performed as follows. Only two input signals were made non-zero. Let us call them T_1 and T_2. Current T_1 was made constant and equal to T_{IN}, while current T_2 changed in a pulsed between values $T_{IN} - 0.5\Delta T_{IN}$ and $T_{IN} + 0.5\Delta T_{IN}$, as shown in Fig. 3.13(a). The pulse starts at time t_{o1} and ends at time t_{o2}. Waveforms y_1 and y_2 have the shape depicted in Fig. 3.13(b). Four different delay times were measured. For the system response caused by a rising edge in T_2, time t_{d1} is the delay between time t_{o1} and the instant at which voltage y_2 crosses the 50% value of its range. Delay t_{d2} is the same for output voltage y_1. For the system response caused by a falling edge in T_2, time t_{d3} is the delay between time t_{o2}

Technology	number of used chips	T_P			
		10μA	100μA	500μA	1mA
ES2_1.0μm	1	2.00%	1.07%	0.58%	0.56%
ES2_1.0μm	2	2.35%	1.03%	0.59%	0.57%
MIETEC_2.4μm	1	1.94%	0.98%	0.70%	0.69%
MIETEC_2.4μm	2	2.15%	1.17%	0.96%	0.87%
MIETEC_2.4μm ES2_1.0μm	2	2.24%	1.05%	0.73%	0.74%

Table 3.2. Current Mode WTA Precision Measurements

Technology	number of used chips	v_P			
		1.5V	2.5V	3.5V	4.5V
ES2_1.0μm	1	0.39%	0.40%	0.41%	1.73%
MIETEC_2.4μm ES2_1.0μm	2	2.62%	9.65%	13.1%	out of range

Table 3.3. WTA Precision Computations (obtained through Hspice simulations) for the Circuit reported in [3.12]

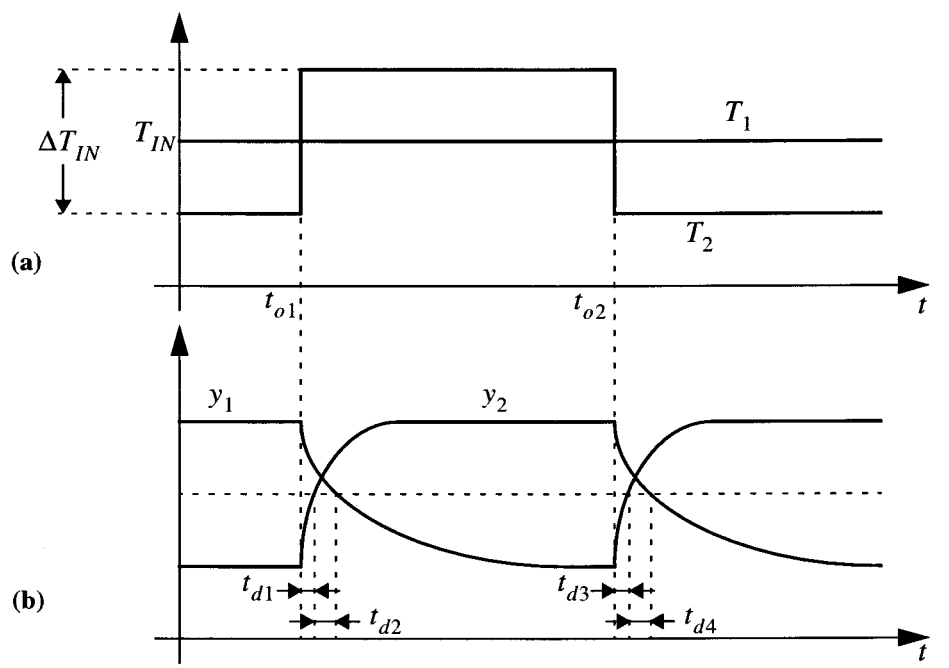


Fig. 3.13: (a) Input Signals, (b) Output Waveforms

T_{IN}	ΔT_{IN}	ES2_1.0μm				MIETEC_2.4μm			
		t_{d1}	t_{d2}	t_{d3}	t_{d3}	t_{d1}	t_{d2}	t_{d3}	t_{d4}
10μA	2μA	6.132μs	4.824μs	2.616μs	3.282μs	3.796μs	2.988μs	3.842μs	5.208μs
10μA	10μA	1.574μs	1.369μs	1.258μs	1.408μs	2.076μs	1.818μs	1.977μs	2.284μs
50μA	10μA	1.289μs	1.023μs	677ns	805ns	1.037μs	909ns	1.058μs	1.178μs
50μA	50μA	364ns	319ns	308ns	342ns	502ns	471ns	455ns	532ns
100μA	20μA	943ns	801ns	222ns	254ns	594ns	587ns	641ns	637ns
100μA	100μA	191ns	167ns	131ns	153ns	276ns	249ns	273ns	281ns
500μA	100μA	161ns	147ns	125ns	128ns	166ns	144ns	112ns	130ns
500μA	200μA	59ns	68ns	48ns	9ns	104ns	95ns	108ns	102ns

Table 3.4. Measured delay times for one chip WTAs

T_{IN}	ΔT_{IN}	ES2_1.0μm			
		t_{d1}	t_{d2}	t_{d3}	t_{d3}
10μA	2μA	16.8μs	6.50μs	5.40μs	3.40μs
10μA	10μA	3.2μs	2.35μs	1.86μs	1.80μs
100μA	20μA	470ns	480ns	750ns	590ns
100μA	100μA	235ns	230ns	270ns	240ns
500μA	100μA	154ns	134ns	375ns	150ns
500μA	200μA	150ns	104ns	136ns	110ns

Table 3.5. Measured delay times for a two-chips WTA

and the instant at which voltage y_1 crosses the 50% value of its range. Delay t_{d4} is the same for output voltage y_2.

Measurements were performed for T_{IN} values of $10\mu A$, $50\mu A$, $100\mu A$, and $500\mu A$, and for ΔT_{IN} equal to $0.2T_{IN}$ and T_{IN}. Table 3.4 shows the measured delay times for those cases where the system is inside one single chip. Table 3.4 shows the delay times measured when a WTA is assembled using 2 chips of the ES2 1.0μm process.

3.7. References

[3.1] T. Serrano-Gotarredona and B. Linares-Barranco, "A High-Precision Current-Mode WTA-MAX Circuit with Multi-Chip Capability," *IEEE Journal of Solid State Circuits*, submitted for publication.

[3.2] T. Serrano and B. Linares-Barranco, "The Active-Input Regulated-Cascode Current Mirror," *IEEE Trans. Circuits*

and Systems-I: Fundamental Theory and Applications, vol. 41, No. 6, pp. 464-467, June 1994.

[3.3] S. Haykin, *Neural Networks: A Comprehensive Foundation*, IEEE Press and Macmillan College Publishing Co., 1994.

[3.4] E. Sánchez-Sinencio and C. Lau, *Artificial Neural Networks: Paradigms, Applications, and Hardware Implementations,* IEEE Press, 1992.

[3.5] J. C. Bezdec and S. K. Pal, *Fuzzy Models for Pattern Recognition*, IEEE Press, 1992.

[3.6] S. A. Elias and S. Grossberg, "Pattern Formation, Contrast Control, and Oscillations in the Short Term Memory of Shunting On-Center Off-Surround Networks, *Biological Cybernetics,* 20, pp. 69-98, 1975.

[3.7] T. Kohonen, *Self-Organization and Associative Memory,* 3rd-ed., Berlin: Springer-Verlag, 1989.

[3.8] A. L. Yuille and D. Geiger, "Winner-Take-All-Mechanisms," in *The Handbook of Brain Theory and Neural Networks,* M. A. Arbib (Ed.), MIT Press, 1995, pp. 1056-1060.

[3.9] Y. He, U. Cilingiroglu, and E. Sánchez-Sinencio, "A High-Density and Low-Power Charge-Based Hamming Network," *IEEE Trans. on VLSI Systems*, vol. 1, No. 1, pp. 56-62, March 1993.

[3.10] B. Linares-Barranco, E. Sánchez-Sinencio, A. Rodríguez-Vázquez, and J. L. Huertas, "A Modular T-Mode Design Approach for Analog Neural Network Hardware Implementations," *IEEE J. Solid-State Circuits,* vol. 27, No. 5, pp. 701-713, May 1992.

[3.11] J. Lazaro, R. Ryckebusch, M. A, Mahowald, C. A. Mead, "Winner-Take-All Networks of O(N) complexity," *Advances in Neural Information Processing Systems*, vol.1, 1989, pp. 703-711.

[3.12] J. Choi, B. J. Sheu, "A High-Precision VLSI Winner-Take-All Circuit for Self-Organizing Neural Networks," *IEEE Journal of Solid State Circuits*, vol. 28, No. 5, May 1993.

[3.13] T. Serrano-Gotarredona and B. Linares-Barranco, "A Real-Time Clustering Microchip Neural Engine," *IEEE Trans. on VLSI Systems,* June 1996.

[3.14] D. G. Nairn And C. A. T. Salama, "Current-Mode Algorithmic Analog-to-Digital Converters," *IEEE Journal of Solid-State Circuits,* vol. 25, August 1990, pp. 997-1004.

[3.15] F. Medeiro-Hidalgo, R. Domínguez-Castro, A. Rodríguez-Vázquez, and J. L. Huertas, "A Prototype Tool for Optimum Analog Sizing Using Simulated Annealing," *Proceeding of the 1992 IEEE Int. Symposium on Circuits and Systems* (ISCAS'92), San Diego, June 1992, vol. 4, pp. 1933-1936.

[3.16] M. J. M. Pelgrom, A. C. J. Duinmaijer, A. P. G. Welbers, "Matching Properties of MOS Transistors," *IEEE Journal of Solid-State Circuits*, vol. 24, No. 5, pp. 1433-1440, 1989.

[3.17] D. Sackinger and W. Guggenbuhl, "A High-Swing, High-Impedance MOS Cascode Circuit," *IEEE J. Solid-State Circuits,* vol. 25, No. 1, pp. 289-298, February 1990.

[3.18] M. G. Degrauwe, J. Rijmenants, E. A. Vittoz, and H. J. De Man, "Adaptive Biasing CMOS Amplifiers," *IEEE Journal of Solid-State Circuits,* vol. SC-17, No. 3, June 1982, pp. 522-528.

[3.19] P. E. Allen and D. R. Holberg, *CMOS Analog Design*, Holt-Rinehart and Winston Inc., New York, 1987.

[3.20] K. Bult and G. J. G. M. Geelen, "The CMOS Gain-Boosting Technique," *Analog Integrated Circuits and Signal Processing,* 1, pp. 119-135, 1991.

[3.21] D. G. Nairn and A. T. Salama, "A Ratio-Independent Algorithmic Analog-to-Digital Converter Combining Current Mode and Dynamic Techniques," *IEEE Trans. Circuits and Systems,* vol. 37, No. 3, pp. 319-325, March 1990.

[3.22] T. Serrano and B. Linares-Barranco, "A Modular Current-Mode High-Precision Winner-Take-All Circuit," *IEEE Trans. Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 42, No. 2, pp. 132-134, February 1995.

Appendix 4: Systematic CMOS Transistor Mismatch Characterization

4.1. Introduction

Mismatching is a limiting factor in circuit design, specially as the transistor geometries are being reduced. Therefore, mismatch characterization is becoming an important task for circuit designers to maintain a proper circuit performance without wasting an excess of circuit area.

The electrical parameters of transistors fabricated in the same die suffer from two kinds of mismatches: a systematic gradient-based deviation and a random deviation with respect to their nominal value. These deviations depend on their position in the wafer, their sizes, and the separation between them.

Suppose we have a die with a two-dimensional array of identical MOS transistors and we measure a certain electrical parameter (for example, the threshold voltage V_T) for each transistor. Fig. 4.1 represents a typical measurement result: horizontal axes x and y represent transistor position in the die, and vertical axis z shows the electrical parameter value. In a typical case we would see a surface that fits the measured values, plus noisy deviations for each point in the surface. The surface represents the systematic gradient-based deviations with respect to the nominal value (or mean value) for all transistors in the same wafer, while the noisy deviations at each point reflect the random deviations.

In this Appendix we will first give the theoretical model of Pelgrom for transistor mismatch properties [4.1], then we will present a chip intended for systematic measurements of transistor mismatch, and finally provide measurement and characterization results.

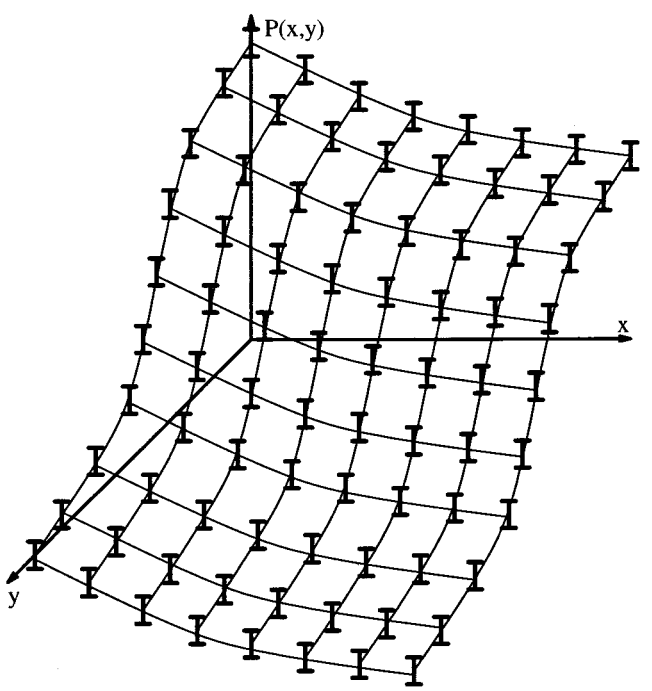


Fig. 4.1: Typical Transistor Property Measurement Result along a Die

4.2. Pelgrom's Model of Transistor Mismatch

If we assume that the electrical property P of a transistor is the result of averaging a certain continuous density function $P(x, y)$ over the transistor area [4.1], the value of parameter P of the transistor of size $W \times L$, with its center point located at position (x_1, y_1), is

$$P_1(x_1, y_1) = \frac{1}{WL} \int\int_{area\,(x_1, y_1)} P(x', y')\, dx'dy' \tag{4.1}$$

The density function $P(x', y')$ contains the variation due to wafer gradients as well as the variation due to the noisy deviations around the interpolated surface.

Under these assumptions, the mismatch in property P between a pair of transistors, sized $W \times L$, located at positions (x_1, y_1) and (x_2, y_2), respectively (as shown in Fig. 4.2), is given by

$$\begin{aligned} \Delta P(x_{12}, y_{12}) &= P_1(x_1, y_1) - P_2(x_2, y_2) = \\ &= \frac{1}{WL} \int\int_{area\,(x_1, y_1)} P(x', y')\, dx'dy' - \frac{1}{WL} \int\int_{area\,(x_2, y_2)} P(x', y')\, dx'dy' = \\ &= \frac{1}{WL} \int\int_{\Re^2} G(x_{12} - x', y_{12} - y')\, P(x', y')\, dx'dy' \end{aligned} \tag{4.2}$$

where, we have denoted as x_{12} and y_{12} the coordinates x and y of the middle point between both transistor centers, that is,

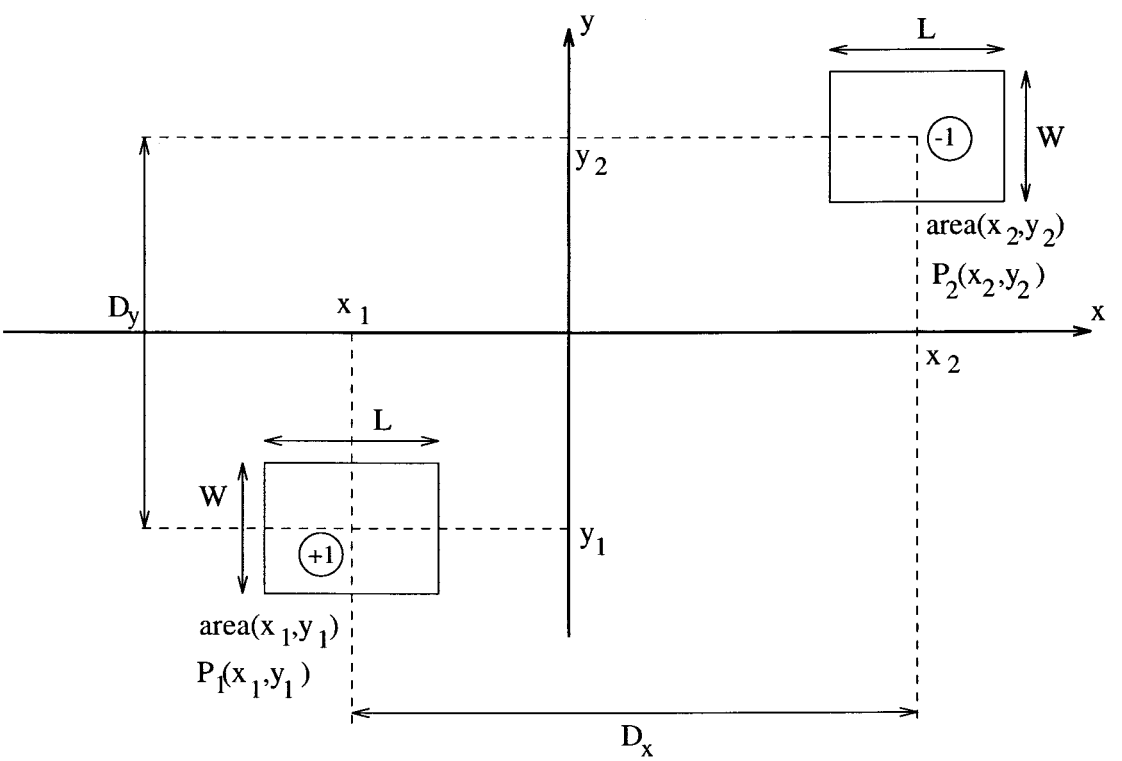


Fig. 4.2: Position and Coordinates of two Transistors

$$\begin{aligned} x_{12} &= \frac{x_1 + x_2}{2} \\ y_{12} &= \frac{y_1 + y_2}{2} \end{aligned}, \tag{4.3}$$

and $G(x, y)$ is a function defined as

$$G(x,y) = \begin{cases} -1 & \text{if } \left(\frac{-D_x - L}{2} \le x \le \frac{-D_x + L}{2}\right) \text{ and } \left(\frac{-D_y - W}{2} \le y \le \frac{-D_y + W}{2}\right) \\ 1 & \text{if } \left(\frac{D_x - L}{2} \le x \le \frac{D_x + L}{2}\right) \text{ and } \left(\frac{D_y - W}{2} \le y \le \frac{D_y + W}{2}\right) \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

$G(x, y)$ has the particularity of being a geometry function, that is, it depends only on the geometries and location of the transistors but not on the parameter density function $P(x, y)$, so it can be computed for each transistor disposition independently of the electrical property.

Taking the Fourier Transform in eq. (4.2) yields,

$$\Delta\mathcal{P}(\omega_x, \omega_y) = \frac{1}{WL}\mathcal{G}(\omega_x, \omega_y)\,\mathcal{P}(\omega_x, \omega_y) \tag{4.5}$$

where $\Delta\mathcal{P}(\omega_x, \omega_y)$ is the Fourier Transform of $\Delta P(x_{12}, y_{12})$, $\mathcal{G}(\omega_x, \omega_y)$ is the one of $G(x, y)$, and $\mathcal{P}(\omega_x, \omega_y)$ is the one of $P(x, y)$.

For the layout of Fig. 4.2 it can be shown that

$$\mathcal{G}(\omega_x, \omega_y) = \frac{\sin\left(\frac{\omega_x L}{2}\right)}{\frac{\omega_x}{2}} \frac{\sin\left(\frac{\omega_y W}{2}\right)}{\frac{\omega_y}{2}} (-2j) \sin\left(\frac{\omega_x D_x + \omega_y D_y}{2}\right) \tag{4.6}$$

If $D_y = 0$, it follows that

$$\left|\frac{1}{WL}\mathcal{G}(\omega_x, \omega_y)\right| = \frac{\sin\left(\frac{\omega_x L}{2}\right)}{\frac{\omega_x L}{2}} \frac{\sin\left(\frac{\omega_y W}{2}\right)}{\frac{\omega_y W}{2}} \left\{2\sin\left(\frac{\omega_x D_x}{2}\right)\right\} \tag{4.7}$$

For a pair of transistors in a common centroid configuration, as shown in Fig. 4.3, it would be

$$\left|\frac{1}{2WL}\mathcal{G}(\omega_x, \omega_y)\right| = \frac{\sin\left(\frac{\omega_x L}{2}\right)}{\frac{\omega_x L}{2}} \frac{\sin\left(\frac{\omega_y W}{2}\right)}{\frac{\omega_y W}{2}} \left\{2\sin\left(\frac{\omega_x D_x}{2}\right)\sin\left(\frac{\omega_y D_y}{2}\right)\right\} \tag{4.8}$$

Fig. 4.4 shows a wafer in which typical contour lines of constant property P have been drawn. In the wafer, at coordinate (x_{12}, y_{12}) a pair of transistors is drawn. Assuming that when averaging $\Delta P(x_{12}, y_{12})$ all over the wafer we have

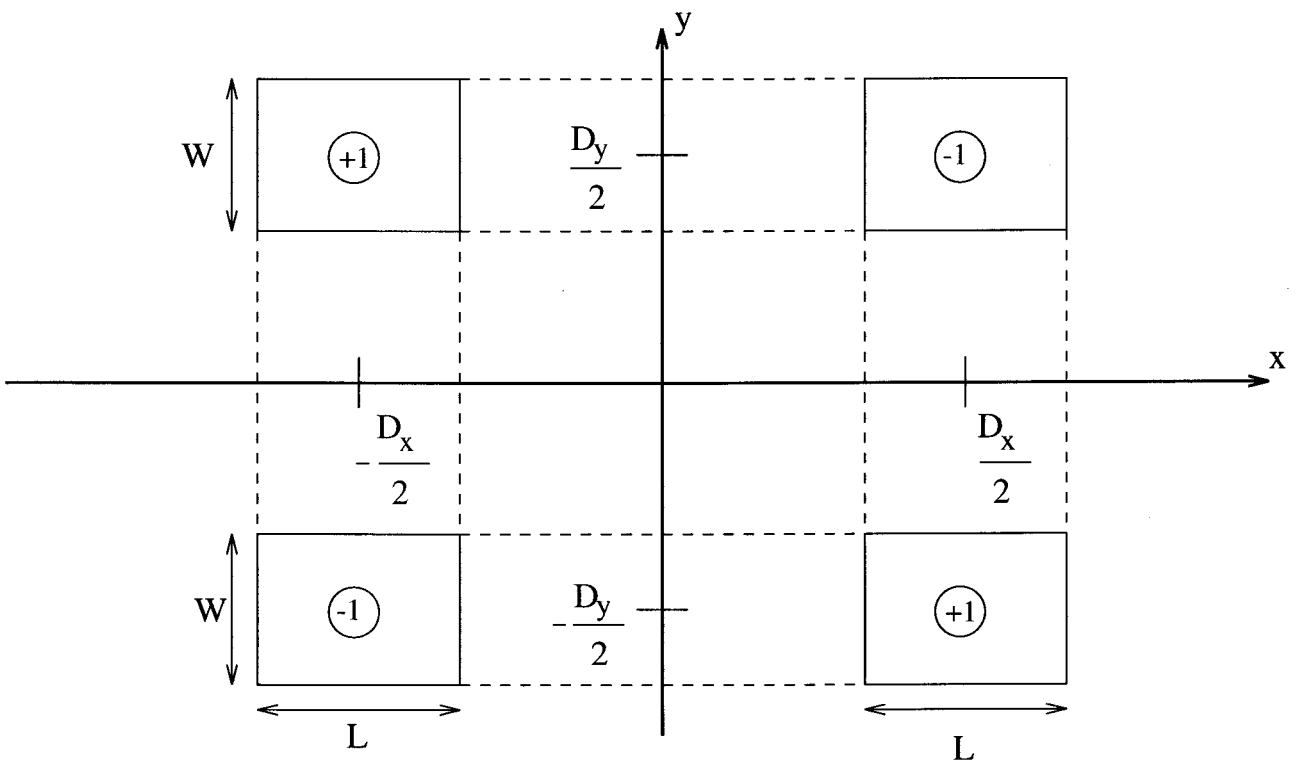


Fig. 4.3: Layout Configuration for a Transistor Pair using Common Centroid

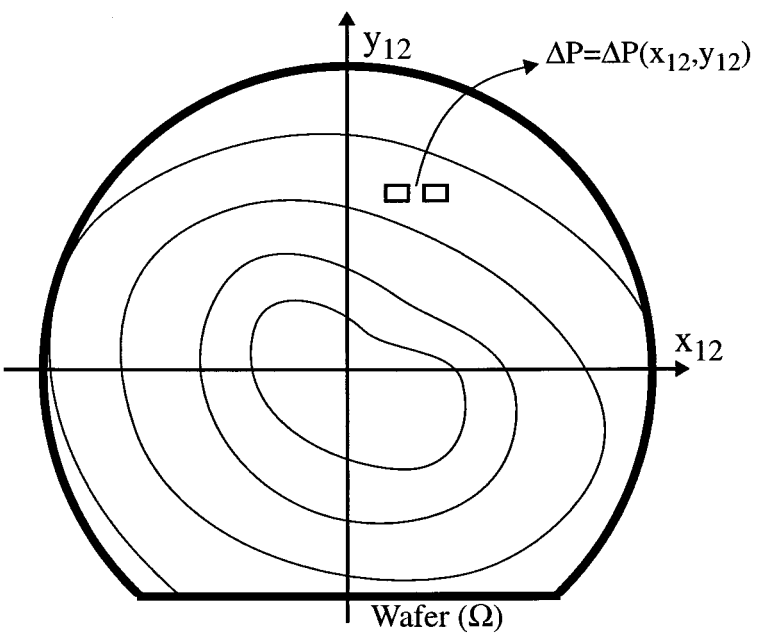


Fig. 4.4: Wafer Gradients

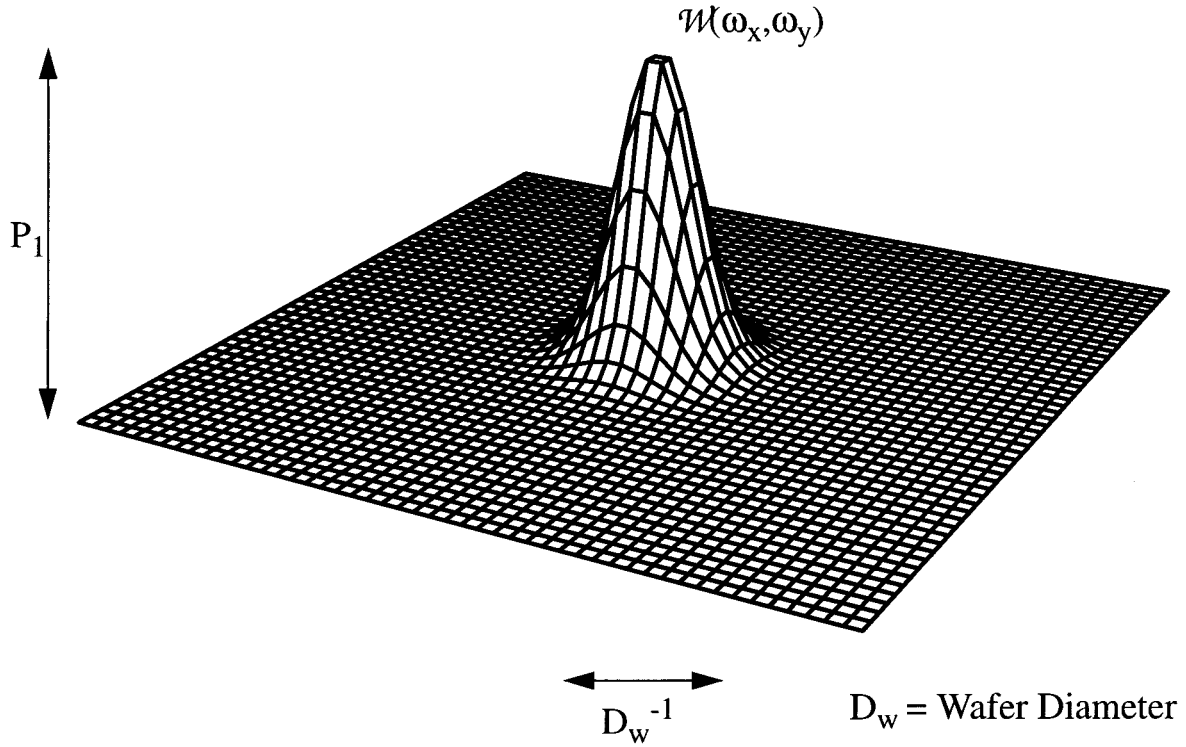


Fig. 4.5: Approximate Shape of Frequency Domain Function $\mathcal{W}(\,)$

$$\overline{\Delta P}\big|_{Wafer} \approx 0 \tag{4.9}$$

we can write that

$$\sigma^2(\Delta P) = \frac{1}{\Omega}\iint_{\Omega} \Delta P^2(x_{12}, y_{12})\, dx_{12} dy_{12} \tag{4.10}$$

where Ω is the area of the wafer. Applying Poisson's Theorem to eq. (4.10) results in

$$\sigma^2(\Delta P) = \frac{1}{4\pi^2\Omega}\int_{-\infty}^{\infty} d\omega_x \int_{-\infty}^{\infty} d\omega_y \left|\frac{1}{WL}\mathcal{G}(\omega_x, \omega_y)\,\mathcal{P}(\omega_x, \omega_y)\right|^2 \tag{4.11}$$

Let us now make the following assumption for function $\mathcal{P}(\omega_x, \omega_y)$:

$$\mathcal{P}(\omega_x, \omega_y) = P_o + \mathcal{W}(\omega_x, \omega_y) \tag{4.12}$$

where P_o is a constant (frequency independent) representative of white noise and $\mathcal{W}(\omega_x, \omega_y)$ is a wafer map component responsible for long distance gradients along the wafer. The spatial frequency content of function $\mathcal{W}(\omega_x, \omega_y)$ is for frequencies of the order of D_w^{-1}, where D_w is the wafer diameter. Therefore, function $\mathcal{W}(\omega_x, \omega_y)$ can be assumed to have a shape of the type depicted in Fig. 4.5, and consequently we can assume that

$$\mathcal{W}(\omega_x, \omega_y) = \begin{cases} \sim P_1 & \text{if } \left(\frac{-1}{D_w} \le \omega_x \le \frac{1}{D_w}\right), \left(\frac{-1}{D_w} \le \omega_y \le \frac{1}{D_w}\right) \\ 0 & \text{otherwise} \end{cases} \tag{4.13}$$

Therefore, eq. (4.11) can be written as

$$\sigma^2(\Delta P) = \frac{1}{4\pi^2 \Omega W^2 L^2} \int_{-\infty}^{\infty} d\omega_x \int_{-\infty}^{\infty} d\omega_y |\mathcal{G}|^2 |P_o + \mathcal{W}|^2 = \\ = \frac{1}{4\pi^2 \Omega W^2 L^2} \{|P_o|^2 Y_1 + Y_2\} \tag{4.14}$$

Assuming a transistor pair as in Fig. 4.2, it would be

$$Y_1 = \int_{-\infty}^{\infty} d\omega_x d\omega_y |\mathcal{G}|^2 = 8\pi^2 WL \tag{4.15}$$

and

$$Y_2 = \int_{-\infty}^{\infty} d\omega_x \int_{-\infty}^{\infty} d\omega_y |\mathcal{G}|^2 \left[P_o^* \mathcal{W} + P_o \mathcal{W}^* + |\mathcal{W}|^2\right] \tag{4.16}$$

Since $D_w \gg D_x, W, L$ then $\mathcal{G}(\omega_x, \omega_y) \approx \omega_x D_x LW$. Thus,

$$Y_2 \approx \int_{-\frac{1}{D_w}}^{\frac{1}{D_w}} d\omega_x \int_{-\frac{1}{D_w}}^{\frac{1}{D_w}} d\omega_y \omega_x^2 D_x^2 L^2 W^2 \left[P_o^* \mathcal{W} + P_o \mathcal{W}^* + |\mathcal{W}|^2\right] = D_x^2 L^2 W^2 k_o' \\ k_o' = \int_{-\frac{1}{D_w}}^{\frac{1}{D_w}} d\omega_x \int_{-\frac{1}{D_w}}^{\frac{1}{D_w}} d\omega_y \omega_x^2 \left[P_o^* \mathcal{W} + P_o \mathcal{W}^* + |\mathcal{W}|^2\right] \tag{4.17}$$

where $P_o^* \mathcal{W} + P_o \mathcal{W}^* + |\mathcal{W}|^2 \approx P_o^* P_1 + P_o P_1^* + |P_1|^2 = k_o$, and,

$$Y_2 = \frac{4 k_o D_x^2 L^2 W^2}{3 D_w^4} \tag{4.18}$$

This results in

$$\sigma^2(\Delta P) = \frac{2|P_o|^2}{\Omega WL} + \frac{k_o D_x^2}{3\pi^2 \Omega D_w^4} = \frac{A_P^2}{WL} + S_P^2 D_x^2 \\ A_P^2 = \frac{2|P_o|^2}{\Omega} \quad , \quad S_P^2 = \frac{k_o}{3\pi^2 \Omega D_w^4} \tag{4.19}$$

For a common centroid configuration it can be shown that the result is

$$\sigma^2(\Delta P) = \frac{|P_o|^2}{\Omega WL} + \frac{k_o D_x^2 D_y^2}{36\pi^2 \Omega D_w^6} = \frac{A_P^2}{2WL} + S_P^2 \frac{D_x^2 D_y^2}{D_w^2} \tag{4.20}$$

As deduced from Pelgrom's Model, the standard deviation (σ) which characterizes the mismatch of parameter P between two transistors of area $W \times L$ separated a distance D is given by

$$\sigma^2(\Delta P) = \frac{A_P^2}{WL} + S_P^2 D^2 \tag{4.21}$$

where A_P and S_P are process dependent parameters that need to be characterized.

In eq. (4.21) we can distinguish two components of the mismatch of electrical parameters between two transistors: an area dependent component (caused by the random deviations of Fig. 4.1) and a distance dependent component (caused by the "randomness" of the location of the transistor pair in the systematic surface of Fig. 4.1).

Parameter A_P stays quite stable from die to die, wafer to wafer, run to run, and even (in first approximation) from foundry to foundry. It can be characterized with a few dies. On the other hand, parameter S_P is more characteristic of each foundry, and for a good characterization many dies per wafer and run are needed. However, parameter S_P is not a critical parameter for circuit designers, since the surface component of Fig. 4.1 can be drastically reduced using layout techniques, such as common centroid (Fig. 7.2-5 of [4.2]) as shown in eq. (4.20), for those transistors whose matching is critical.

In this Appendix we provide characterization results for the parameters A_P. These parameters which cannot be compensated with layout techniques, and therefore need to be well characterized so that circuit designers can take their effects into account during the circuit design process.

4.3. Mismatch Characterization Chip

For the mismatch characterization of each CMOS process we propose to fabricate a special purpose chip for each technology. In this Appendix we will show the characterization results of parameter A_P of eq. (4.21) for two CMOS processes: the ES2[1] 1.0μm CMOS process and the CNM[2] 2.5μm CMOS process. For each

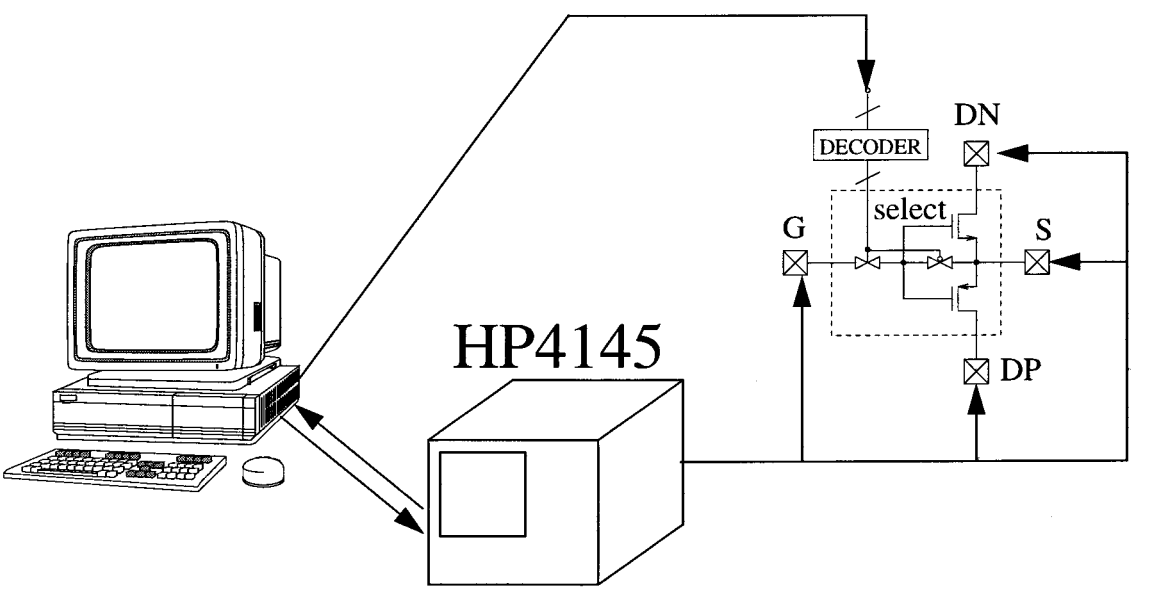


Fig. 4.6: Experimental montage for the Automatic Characterization of the "Mismatching" between MOS Transistors

process a chip was fabricated that contains a matrix of cells. Each cell contains a number of NMOS and PMOS transistors of different sizes. Additional decoding-selection circuitry is added to each cell and to the chip, so that only one transistor at a time is selected and connected to the outside pins for characterization. This way, we can bypass the procedure of accessing each transistor with special probes, reducing significantly the measurement time and cost.

Fig. 4.6 shows a simplified diagram of the chip and the external control and measurement equipment. In the chip, transistors are grouped by pairs: one NMOS and one PMOS. All NMOS transistors in the chip share their Drains in a common node connected to the pin DN. All PMOS transistors share their Drains at pin DP. All NMOS and PMOS transistors share their Sources at pin DS. All the transistors have their Gates short-circuited to their Sources, except for one transistor pair: the one that receives a high "select" signal. This pair has their Gates connected to the external pin G. If pin DP is left unconnected and the current between pins S and DN is measured, then the selected NMOS transistor is being measured. If pin DN is left unconnected and one measures the current between pins S and DP, then the selected PMOS transistor is being measured. Fig. 4.7 shows the schematic of the circuitry in a chip containing a matrix of 8×8 cells[3] with 8 pairs of NMOS and PMOS transistors of different sizes inside each cell. For this chip we can select separately each of the 512 transistor pairs using a decoding circuitry with 8 control bits. One group of bits (from b_0 to b_1 in Fig. 4.7) selects the active matrix row through the row decoder. Another group of bits (from b_2 to b_4 in Fig. 4.7) selects the active matrix column through the column decoder. Finally a decoder which selects the size of the active transistor pair inside the selected cell is added to each cell and is controlled by another group of bits (from b_5 to b_7 in Fig. 4.7).

The active pair of transistors is selected by the digital decoding circuitry controlled through an external bus by a host computer. The host computer also controls a DC curve tracer (HP4145) which through chip pins DN, DP, S and G measures the curves of the active NMOS and PMOS transistors. By this way, it is possible to characterize a large number of transistors per chip without automatic probe positioning machines.

4.4. Transistor Measurement

The most critical electrical parameters responsible for current mismatches between transistors are:

- Beta: $\beta = \frac{W}{L}\mu C_{ox}$
- Threshold Voltage: V_{T0}
- Gamma: γ

In a Level 1 (H)Spice model of the MOS transistor, its Drain to Source current is given by

1. ES2: European Silicon Structures, available through the EUROPRACTICE services.
2. CNM: National Microelectronics Center Silicon Foundry at Barcelona, Spain.
3. For CNM-2.5μm technology the chip contains a 7×8 cell array, and for the ES2-1.0μm technology the chip contains an 8×8 array of cells.

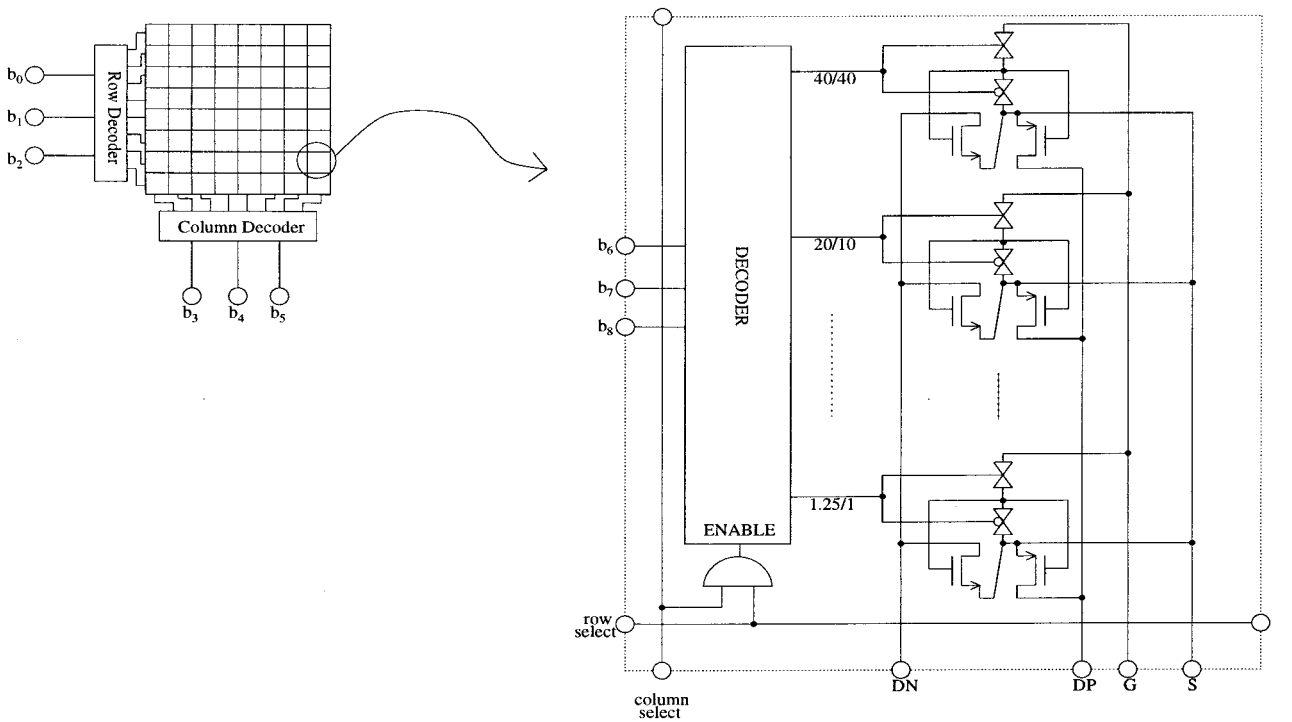


Fig. 4.7: Schematic of the internal decoding circuitry in the chip

$$I_{DS} = \beta \frac{(V_{GS} - V_T(V_{SB}))^2}{1 + \theta (V_{GS} - V_T(V_{SB}))} \qquad V_{DS} \geq V_{GS} - V_T$$

$$I_{DS} = \beta \frac{\left(V_{GS} - V_T(V_{SB}) - \frac{1}{2}V_{DS}\right)V_{DS}}{1 + \theta (V_{GS} - V_T(V_{SB}))} \qquad V_{DS} \leq V_{GS} - V_T \tag{4.22}$$

where θ is responsible for the mobility degradation effect. The threshold voltage, which depends on the Source to Bulk voltage, is given by,

$$V_T(V_{SB}) = V_{T0} + (\eta - 1) V_{SB} + \gamma\left(\sqrt{\phi + V_{SB}} - \sqrt{\phi}\right) \tag{4.23}$$

where $\phi = 0.6V$, η is an extra (fitting) parameter, and $V_{T0} = V_T(V_{SB}=0)$.

For each transistor two curves are measured:

- Curve 1: $V_{DS} = 0.1V$, $V_{SB} = 0V$

  $V_{GS} = 1.5V\text{-}5.0V$

  $$I_{DS} = \beta \frac{\left(V_{GS} - V_{T0} - \frac{1}{2}0.1V\right)0.1V}{1 + \theta (V_{GS} - V_{T0})}$$

- Curve 2: $V_{DS} = 0.1V$

  $V_{GS} = 3.0V$

$$V_{SB} = 0.0V\text{-}2.0V$$

$$I_{DS} = \beta \frac{\left(3.0V - V_T(V_{SB}) - \frac{1}{2}0.1V\right)0.1V}{1 + \theta(3.0V - V_T(V_{SB}))}$$

For Curve 1 parameters β, V_{T0}, and θ are fitted using Nonlinear Curve Fitting [4.3]. Random deviations in parameters θ and η are assumed to have negligible contribution to current mismatches, so parameter θ is only fitted for the first transistor of each size and that value is taken for the other transistors of the same size in the same chip. This way, the current mismatch is assumed to be due to mismatches in parameters β and V_{T0} only. For Curve 2, parameters β, V_{T0}, and θ are taken from the fitted values from Curve 1, and the measured curve

$$V_\gamma = V_T(V_{SB}) - V_{T0} = V_{GS} - V_{T0} - \frac{I_{DS} + \frac{1}{2}\beta V_{DS}^2}{\beta V_{DS} - \theta I_{DS}} \quad (V_{DS}=0.1V,\ V_{GS}=3.0V) \tag{4.24}$$

is fitted to the curve

$$V_\gamma = (\eta - 1)V_{SB} + \gamma\left(\sqrt{\phi + V_{SB}} - \sqrt{\phi}\right). \tag{4.25}$$

Parameter η is fitted only for the first transistor of each size and assumed to be the same for the other transistors of the same size. This assumption leaves a random nature only for parameter γ.

In this way, for all the transistors of the same size, parameters β, V_{T0}, and γ are extracted. Using this extraction procedure, we can represent the parameter $P_{W,L}$ measured for a transistor of size $W \times L$ as a function of the position of the transistor in the chip, thus generating the surface $P_{W,L}(x, y)$.

Repeating this procedure for the different sizes and types of transistors we obtain the family of surfaces $\beta_{W,L}(x, y)$, $V_{TO-W,L}(x, y)$ and $\gamma_{W,L}(x, y)$ for the NMOS and PMOS transistors. For technology CNM-2.5 µm 56 identical transistors were available in each chip, and 4 different transistor geometries were available in each cell, for both NMOS and PMOS transistors. For technology ES2-1.0µm there were 196 transistors of each of the 8 different geometries and for NMOS and PMOS types. To illustrate the surfaces we obtained for each parameter and for each set of equal sized transistors in each chip, Fig. 4.8 shows the surfaces $\beta_{W,L}(x, y)$ for the NMOS transistors of the 4 different sizes for a chip of the CNM2.5µm process, and Fig. 4.9 depicts the surfaces $V_{TO-W,L}(x, y)$ for the PMOS transistors of the 8 different sizes for a chip of the ES2 1.0µm process. Each surface represents the measured transistor parameter as a function of the position of the transistor in the chip. Below each surface a contour diagram has been included.

4.5. Statistical Data Processing

Let us name each of the surfaces obtained after applying the extraction procedure in the following way:

$$P_{SK} = P_{SK}(n\Delta x, m\Delta y) \tag{4.26}$$

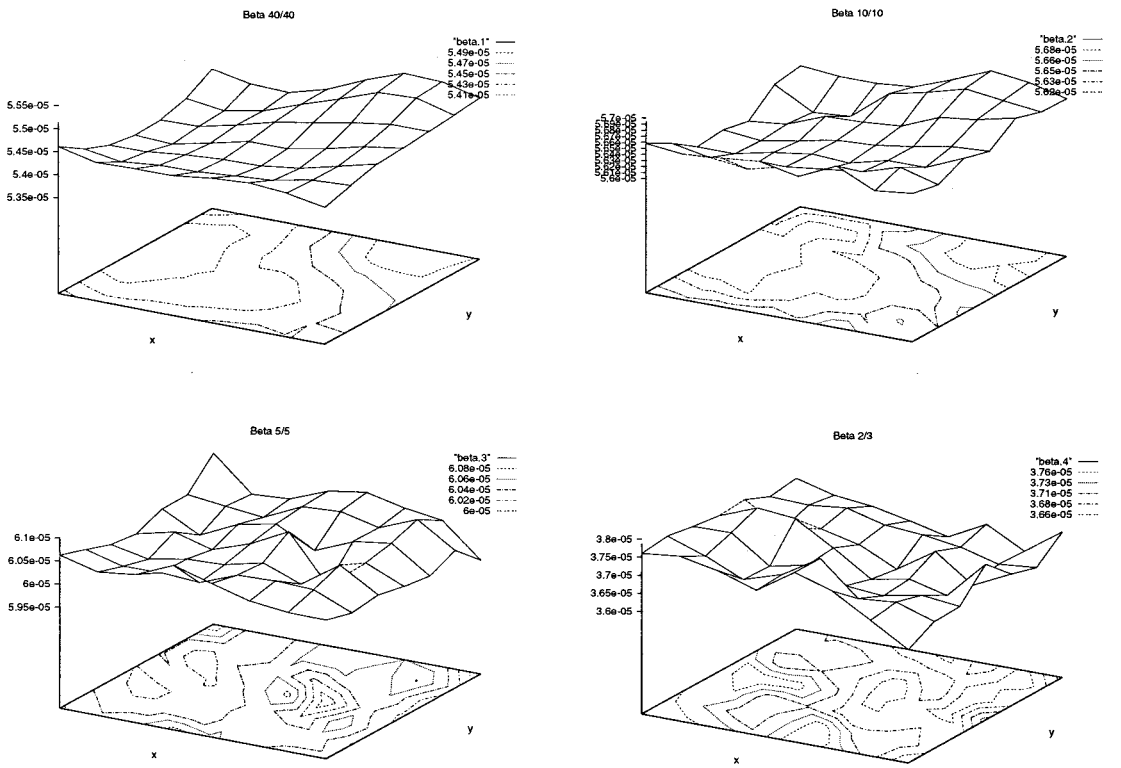


Fig. 4.8: Surfaces β(x,y) for NMOS transistor with size (W=L=40μm), (W=L=10μm), (W=L=5μm), (W=2μm, L=3μm) for a chip of the CNM 2.5μm process

where P is the extracted parameter which can be either β, V_{T0}, and γ, S denotes the transistor size, K the type of transistor ($K \equiv N$ for NMOS, $K \equiv P$ for PMOS), and n and m specify the position of the transistor in the array. Δx and Δy are the separations between two adjacent cells of the array in the x and y directions.

According to this notation, the mean value of an electrical parameter $\overline{P_{SK}}$ in each chip would be computed as follows:

$$\overline{P_{SK}} = \frac{\sum_{n=0}^{n^{max}} \sum_{m=0}^{m^{max}} P(n\Delta x, m\Delta y)}{n^{max} \times m^{max}} \tag{4.27}$$

where n^{max} and m^{max} are the number of times each cell is repeated in the x and y directions.

After computing the mean of each parameter, the following differences are computed at each point,

$$\Delta_x P_{SK}(n,m) = P_{SK}((n+1)\Delta x, m\Delta y) - P_{SK}(n\Delta x, m\Delta y) \quad \begin{cases} n = 0, \ldots n^{max} - 1 \\ m = 0, \ldots m^{max} \end{cases} \tag{4.28}$$

$$\Delta_y P_{SK}(n,m) = P_{SK}(n\Delta x, (m+1)\Delta y) - P_{SK}(n\Delta x, m\Delta y) \quad \begin{cases} n = 0, \ldots n^{max} \\ m = 0, \ldots m^{max} - 1 \end{cases} \tag{4.29}$$

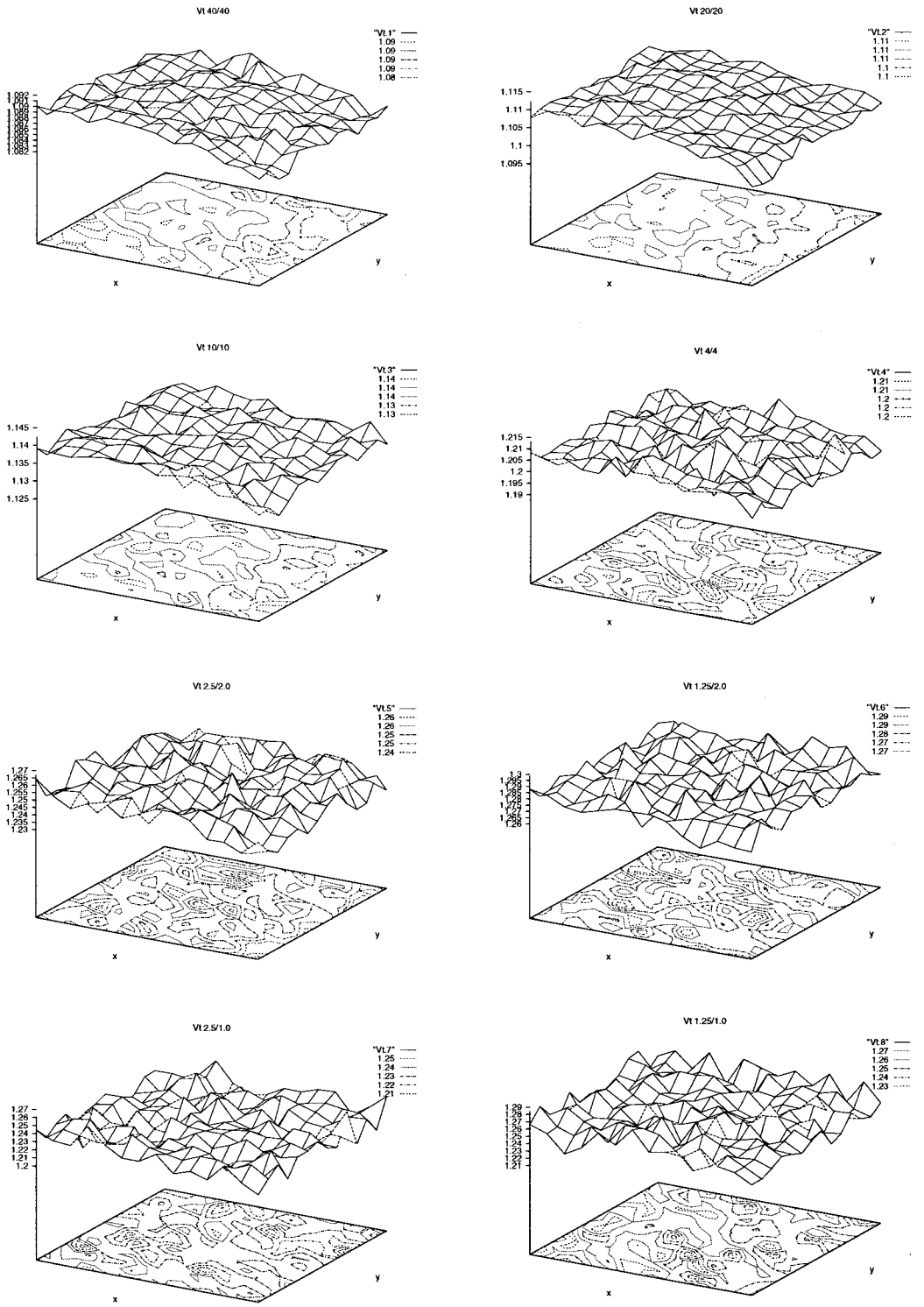


Fig. 4.9: Surfaces $V_{T0}(x,y)$ for PMOS transistor sized (W=L=40μm), (W=L=20μm), (W=L=10μm), (W=L=4μm), (W=2.5μm, L=2μm), (W=2.5μm, L=1μm), (W=1.25μm, L=2μm), (W=1.25μm, L=1μm) for a chip of the ES2 1.0μm process

Afterwards, the standard deviation of the difference between parameters along the *X*-direction is computed:

$$\sigma^2_{PX,\,SK} = \sigma^2(\Delta_x P_{SK}) = \frac{\sum_{n=0}^{n^{max}-1}\sum_{m=0}^{m^{max}}\left(\Delta_x P_{SK}(n,m) - \overline{\Delta_x P_{SK}}\right)^2}{(n^{max}-1)\,m^{max}} \tag{4.30}$$

In the same way, the standard deviation along the *Y*-direction is computed:

$$\sigma^2_{PY,\,SK} = \sigma^2(\Delta_y P_{SK}) = \frac{\sum_{n=0}^{n^{max}}\sum_{m=0}^{m^{max}-1}\left(\Delta_y P_{SK}(n,m) - \overline{\Delta_y P_{SK}}\right)^2}{n^{max}(m^{max}-1)} \tag{4.31}$$

Next, the average between these two quantities is computed, and its square-root, in %, is given as the relative standard deviation:

$$\sigma_{P,\,SK} = \frac{\sqrt{\frac{\sigma^2_{PX,\,SK} + \sigma^2_{PY,\,SK}}{2}}}{\overline{P_{SK}}}100 \tag{4.32}$$

Both, $\sigma_{P,\,SK}$ and $\overline{P_{SK}}$ are computed for each die and for each transistor size.

Assuming these data fit to curves of the type

$$\sigma_{P,\,SK} = \frac{A_P}{\sqrt{W_{eff}L_{eff}}} \qquad \begin{cases} W_{eff} = W - WD \\ L_{eff} = L - LD \end{cases} \tag{4.33}$$

the value of A_P was computed for each parameter P and for each die. Fig. 4.10 depicts the dependence with $1/\sqrt{WL}$ of the measured values of $\sigma(\Delta\beta)$, $\sigma(\Delta V_{TO})$ and $\sigma(\Delta\gamma)$ of the NMOS transistors of one chip of the CNM 2.5µm technology.

Finally, the values of $\sigma_{P,\,SK}$ are averaged over all dies. Table 4.4 shows the computed averaged standard deviations for the chips of the CNM 2.5µm technology[4] and Table 4.4 shows the averaged standard deviatioons for the chips of the ES2-1.0µm technology. These averaged values are also assumed to fit to a curve of the type given by eq. (4.33). The fitted slopes of these curves are the statistical parameters A_P which characterize the technological process. Table 4.3 contains the final values of parameters A_β, $A_{V_{TO}}$ and A_γ obtained fitting the curves over the averaged deviations for the NMOS and PMOS transistors of the CNM-2.5µm and the ES2-1.0µm CMOS processes.

In order to evaluate the confidence of these measurements, for one of the dies one transistor of each size was measured repeatedly (56 times) and the standard deviations were computed as if a 56 elements array was measured. Table 4.4 gives the values computed for the CNM 2.5µm process. These deviations represent the error within which the values of Table 4.4 have been measured and computed.

4. The results were measured for two types of substrates: (a) substrate of type p, and (b) substrate of type p with epitaxial p+ layer (p/p+).

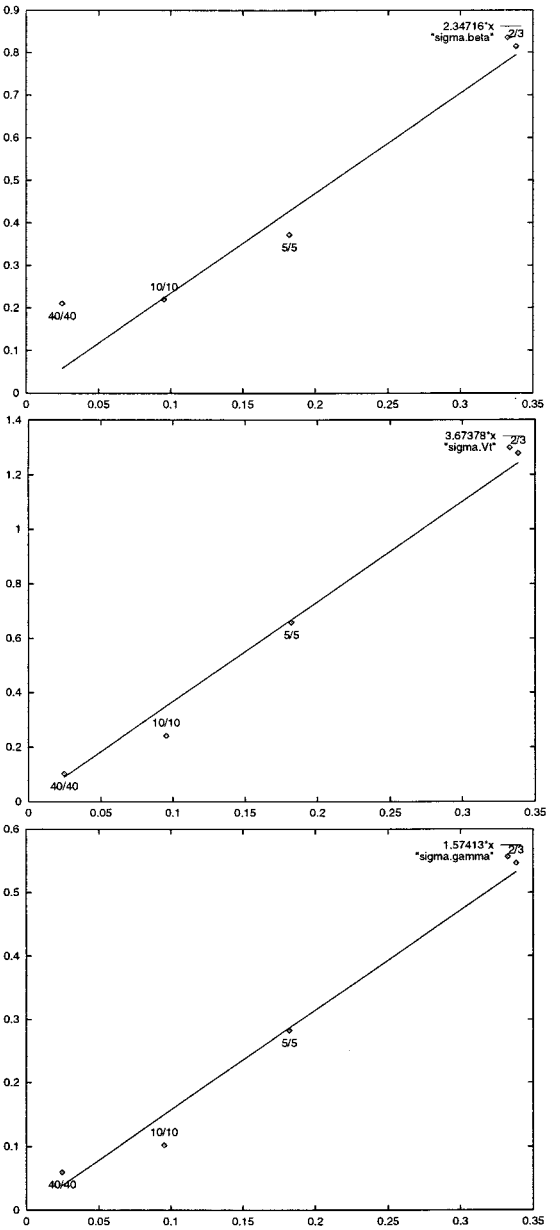


Fig. 4.10: Dependence of functions $\sigma(\Delta\beta)$, $\sigma(\Delta V_{T0})$ and $\sigma(\Delta\gamma)$ (in %) versus $1/\sqrt{WL}$ for NMOS transistors of the CNM 2.5μm technology

		W = 40μm L = 40μm	W = 20μm L = 10μm	W = 10μm L = 10μm	W = 5μm L = 4μm	W = 2.5μm L = 2μm	W = 2.5μm L = 1μm	W = 1.25μm L = 2μm	W = 1.25μm L = 1μm
NMOS	$\sigma(\beta)$ (%)	0.080	0.131	0.310	0.371	0.951	0.910	0.930	1.282
	$\sigma(V_{T0})$ (mV)	0.652	1.339	1.362	2.623	5.218	9.940	6.210	13.122
	$\sigma(\gamma)$ (mV^{-1})	0.396	0.763	0.487	1.415	3.421	5.698	3.037	9.652
PMOS	$\sigma(\beta)$ (%)	0.093	0.120	0.297	0.427	0.711	1.676	0.681	1.574
	$\sigma(V_{T0})$ (mV)	1.201	1.535	1.492	2.787	5.188	11.868	7.132	14.147
	$\sigma(\gamma)$ (mV^{-1})	0.522	0.646	0.697	1.612	4.148	7.011	4.051	7.945

Table 4.2: Standard Deviations Averaged over all Dies for the ES2-1.0μm process

			$W = 40\mu m$ $L = 40\mu m$	$W = 10\mu m$ $L = 10\mu m$	$W = 5\mu m$ $L = 5\mu m$	$W = 2\mu m$ $L = 3\mu m$
NMOS	p	$\sigma(\beta)$	0.2200704	0.2612075	0.3846001	0.979007
		$\sigma(V_{T0})$	0.1236119	0.2850701	0.6125308	1.351129
		$\sigma(\gamma)$	0.0657665	0.1154645	0.2588394	0.5734347
	p/p+	$\sigma(\beta)$	0.3447584	0.3985946	0.5431468	0.8584701
		$\sigma(V_{T0})$	0.2419148	0.4164946	0.9343108	1.512313
		$\sigma(\gamma)$	0.1229629	0.2150784	0.4434690	0.7853522
PMOS	p	$\sigma(\beta)$	0.2194395	0.2921146	0.4232929	1.061984
		$\sigma(V_{T0})$	0.1935054	0.3779017	0.7190964	1.892969
		$\sigma(\gamma)$	0.0707893	0.1269131	0.2446530	0.5855953
	p/p+	$\sigma(\beta)$	0.2477367	0.3357617	0.5007437	0.8304728
		$\sigma(V_{T0})$	0.3599902	0.6305608	0.9422545	1.590794
		$\sigma(\gamma)$	0.1210937	0.3141505	0.6457047	0.8907923

Table 4.1: Standard Deviations Averaged over all Dies for the CNM 2.5μm process

	CNM 2.5μm		ES2 1.0μm	
	NMOS	PMOS	NMOS	PMOS
A_β	2.3	3.1	2.5	2.3
$A_{V_{T0}}$	3.7	6.2	1.6	1.1
A_γ	1.6	1.4	1.2	1.3

Table 4.3. Extracted A_P parameters expressed in %μm

Another interesting statistical data, which may be of great help when designing a circuit, are the correlations that may appear between the different electrical parameters β_n, V_{TO_n}, γ_n, β_p, V_{TO_p} and γ_p, for each transistor size and for each technology. As previously said, we have 56 identical transistors in each CNM-2.5μm chip, and 64 in the ES2-1.0μm chips. Consequently, for each transistor size we can compute the statistical correlations between these 6 extracted parameters. The correlations between parameters P_1 and P_2 for transistors of size $\frac{W}{L}$ of a given chip are

$$r_{W/L}(P_1, P_2) = \frac{\sum_{n=1}^{n^{max} \times m^{max}} \left(P_1(n) - \overline{P_1}\right)\left(P_2(n) - \overline{P_2}\right)}{n^{max} m^{max} \sigma(P_1)\sigma(P_2)}. \quad (4.34)$$

These values are computed for each chip and then averaged over all available chips. Also the standard deviation $\sigma_{W/L}(P_1, P_2)$ of the values that the correlation $r_{W/L}(P_1, P_2)$ takes over the different chips is computed. If a small

	$W = 40\mu m$ $L = 40\mu m$	$W = 10\mu m$ $L = 10\mu m$	$W = 5\mu m$ $L = 5\mu m$	$W = 2\mu m$ $L = 3\mu m$
$\sigma(\beta)$	0.0190286	0.0155667	0.0171943	0.0148482
$\sigma(V_{T0})$	0.0190708	0.0166157	0.0178704	0.023583
$\sigma(\gamma)$	0.0193913	0.0203743	0.022757	0.0260463
$\sigma(\beta)$	0.0182232	0.0213448	0.018121	0.0227736
$\sigma(V_{T0})$	0.0202286	0.0207114	0.0221505	0.0247409
$\sigma(\gamma)$	0.0497218	0.0538003	0.052774	0.0537932

Table 4.4: Confidence of Measured/Computed Deviations of Table 4.4

CNM	β_n-V_{T0_n}	β_n-γ_n	V_{T0_n}-γ_n	β_p-V_{T0_p}	β_p-γ_p	V_{T0_p}-γ_p
40/40	0.16±0.45	-0.50±0.34	0.59±0.40	-0.21±0.44	0.19±0.25	-0.21±0.44
10/10	0.20±0.30	-0.31±0.38	0.67±0.23	-0.31±0.37	0.02±0.32	-0.31±0.37
5/5	0.14±0.29	-0.13±0.40	0.70±0.14	-0.10±0.37	-0.18±0.38	-0.10±0.37
2/3	0.01±0.29	-0.11±0.34	0.65±0.18	-0.28±0.30	-0.22±0.23	-0.28±0.30

Table 4.5: CNM 2.5µm Correlations (mean±sigma)

CNM	β_n-V_{T0_p}	β_n-γ_p	V_{T0_n}-β_p	V_{T0_n}-γ_p	γ_n-β_p	γ_n-V_{T0_p}
40/40	0.09±0.37	0.21±0.33	0.42±0.34	0.48±0.34	0.46±0.29	-0.28±0.46
10/10	0.04±0.29	0.17±0.38	0.25±0.36	0.22±0.33	0.24±0.38	-0.12±0.25
5/5	0.05±0.33	-0.09±0.43	0.10±0.32	0.19±0.34	0.00±0.39	-0.23±0.26
2/3	0.00±0.35	-0.23±0.17	-0.03±0.28	0.10±0.20	-0.12±0.28	-0.14±0.24

Table 4.6: CNM 2.5µm Correlations (mean±sigma)

CNM	β_n-β_p	V_{T0_n}-V_{T0_p}	γ_n-γ_p
40/40	-0.18±0.24	-0.08±0.58	0.28±0.44
10/10	-0.03±0.43	0.01±0.29	0.20±0.32
5/5	0.19±0.48	-0.19±0.28	0.24±0.29
2/3	0.39±0.29	-0.14±0.22	0.11±0.15

Table 4.7: CNM 2.5µm Correlations (mean±sigma)

value of $\sigma_{W/L}(P_1, P_2)$ results, the corresponding correlation has a stable value over all measured chips. Otherwise, the correlations suffer large (random) oscillations from chip to chip. Table 4.5, Table 4.6 and Table 4.7 show the correlations for the CNM-2.5µm process and Table 4.8, Table 4.9 and Table 4.10 show the correlations for the ES2-1.0µm process. The tables show the "mean ± sigma" of the correlations computed over all available chips. For CNM-2.5µm there were 11 operative chips, and for ES2-1.0µm there were 8.

ES2	$\beta_n - V_{TO_n}$	$\beta_n - \gamma_n$	$V_{TO_n} - \gamma_n$	$\beta_p - V_{TO_p}$	$\beta_p - \gamma_p$	$V_{TO_p} - \gamma_p$
40/40	0.34±0.10	0.12±0.26	0.81±0.12	-0.06±0.30	-0.22±0.31	-0.29±0.29
20/10	0.05±0.20	-0.08±0.22	0.62±0.10	-0.37±0.16	+0.14±0.19	-0.28±0.12
10/10	-0.19±0.24	-0.15±0.20	0.54±0.18	-0.18±0.16	-0.03±0.11	-0.19±0.30
5/4	-0.27±0.11	-0.25±0.10	0.43±0.16	-0.30±0.12	-0.04±0.11	-0.27±0.16
2.5/2	-0.33±0.10	-0.46±0.09	0.41±0.11	-0.40±0.15	-0.19±0.12	-0.19±0.13
1.25/2	-0.26±0.90	-0.39±0.11	0.44±0.13	-0.29±0.07	-0.30±0.09	-0.07±0.11
2.5/1	-0.61±0.11	-0.75±0.04	0.66±0.14	-0.59±0.10	-0.68±0.08	0.30±0.13
1.25/1	-0.41±0.08	-0.57±0.08	0.55±0.09	-0.57±0.05	-0.64±0.05	0.27±0.11

Table 4.8: ES2 1.0μm correlations (mean±sigma)

ES2	$\beta_n - V_{TO_p}$	$\beta_n - \gamma_p$	$V_{TO_n} - \beta_p$	$V_{TO_n} - \gamma_p$	$\gamma_n - \beta_p$	$\gamma_n - V_{TO_p}$
40/40	0.09±0.36	-0.24±0.37	0.42±0.16	-0.19±027	0.31±0.27	0.36±0.34
20/10	-0.01±0.26	-0.11±0.19	0.13±0.30	-0.01±0.22	0.16±0.26	-0.06±0.29
10/10	-0.03±0.17	-0.12±0.11	-0.04±0.18	-0.19±0.22	-0.01±0.16	-0.18±0.31
5/4	0.01±0.12	-0.08±0.14	0.02±0.15	-0.05±0.13	0.11±0.13	-0.16±0.23
2.5/2	-0.05±0.09	-0.08±0.12	-0.06±0.10	-0.06±0.13	0.03±0.10	-0.06±0.14
1.25/2	-0.05±0.10	-0.11±0.10	-0.06±0.08	0.06±0.11	-0.06±0.08	-0.01±0.11
2.5/1	-0.25±0.10	-0.32±0.11	-0.32±0.17	-0.32±0.13	-0.40±0.10	0.27±0.09
1.25/1	-0.01±0.14	-0.09±0.13	-0.09±0.13	0.13±0.16	-0.10±0.14	0.04±0.11

Table 4.9: ES2 1.0μm correlations (mean±sigma)

ES2	$\beta_n - \beta_p$	$V_{TOn} - V_{TO_p}$	$\gamma_n - \gamma_p$
40/40	0.54±0.18	0.60±0.28	0.02±0.26
20/10	0.27±0.23	0.25±0.27	0.05±0.23
10/10	0.56±0.08	-0.03±0.29	0.14±0.21
5/4	0.01±0.10	-0.02±0.28	-0.08±0.10
2.5/2	0.02±0.06	-0.12±0.13	-0.06±0.13
1.25/2	0.20±0.10	-0.05±0.11	0.04±0.08
2.5/1	0.38±0.11	0.15±0.11	0.38±0.10
1.25/1	0.03±0.15	0.06±0.10	0.16±0.14

Table 4.10: ES2 1.0μm correlations (mean±sigma)

4.6. (H)Spice Simulations

These results can now be useful for circuit design. By including them in the simulation files a circuit designer can estimate the deviation of the different circuit parameters. This allows proper sizing of transistors and maintain design parameters within acceptable limits without wasting excessive area. Fig. 4.11 shows an input file for simulator (H)Spice where these results are used through Monte Carlo simulations.

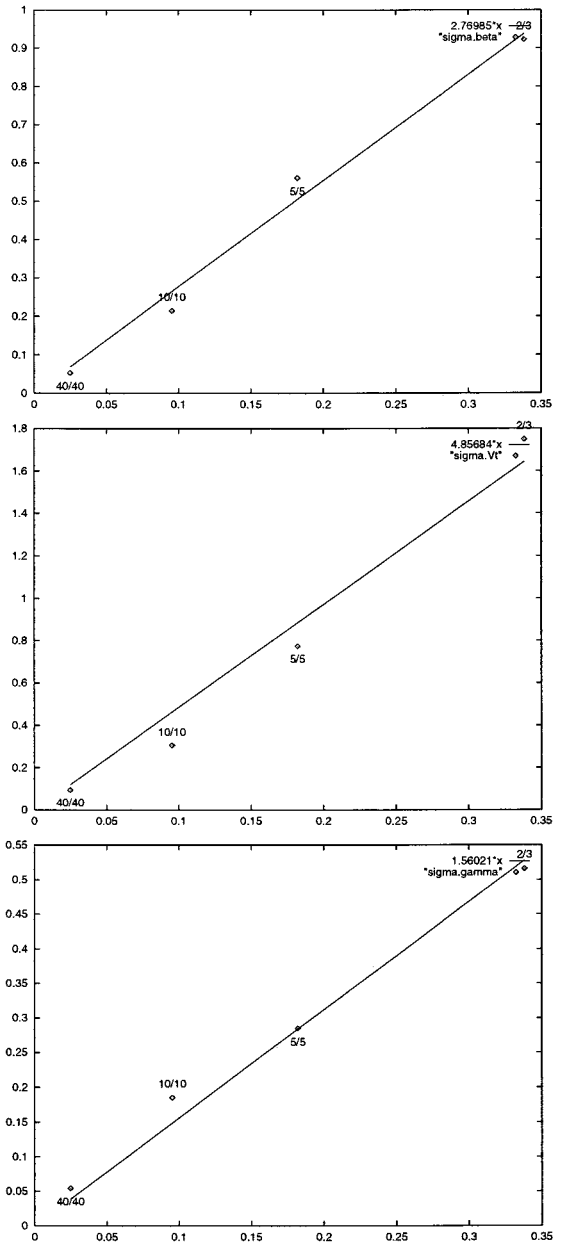


Fig. 4.12: Mismatch Characteristics predicted by Simulation

As a verification exercise, the simulator was used to obtain the same curves that were measured experimentally. Then, the electrical parameters of the transistors were extracted using the same procedure of non-linear curve fitting as in the case of the transistor curves measured experimentally. For the extracted parameter the same computations as explained in Section 4.5 were done to obtain the statistical parameters A_P. Fig. 4.12 shows the mismatch characteristics predicted now by the simulator. As can be seen, they agree very well with the measured data results.

4.7. References

[4.1] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching Properties of MOS Transistors," *IEEE Journal of Solid-State Circuits,* vol. 24, No. 5, pp. 1433-1440, October 1989.

[4.2] P. E. Allen and D. R. Holberg, *CMOS Analog Circuit Design.* New York: Holt, Rinehart and Winston, Inc., 1987.

[4.3] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C. The Art of Computing.*

```
.param kp_n_global=58e-6
.param kp_p_global=16.7e-6
.param vto_n_global=1.00
.param vto_p_global=-1.10
.param gamma_n_global=1.26
.param gamma_p_global=0.70
.subckt nmod Drain Gate Source Bulk width=w length=l
      m_nmod Drain Gate Source Bulk nmod w=w l=l
      + ad='w*4u' as='w*4u' pd='2*w+8u' ps='2*w+8u'
      .MODEL nmod NMOS
      + LEVEL = 2       VTO    = vto.n   KP  = kp.n    GAMMA = gamma.n
      + PHI   = 0.72    LAMBDA = 0.009   MOB = 7       THETA = 0.154
      + NSUB  = 2E16    XJ     = 1E-6    JS  = 0.73E-3 RSH   = 25
      + TOX   = 3.6E-8  LD     = 0.75U   WD  = 0U      PB    = 0.83
      + CJ    = 5.63E-4 CJSW   = 2.7E-9  MJ  = 0.43    MJSW  = 0.3
      + AF    = 1       KF     = 2.3E-27
      .param Ab=2.743179e-8
      .param Av=3.806265e-8
      .param Ag=1.612862e-8
      .param sigma_kp_n='kp_n_global*Ab*sqrt(1/w/l)'
      .param sigma_vto_n='vto_n_global*Av*sqrt(1/w/l)'
      .param sigma_gamma_n='gamma_n_global*Ag*sqrt(1/w/l)'
      .param kp.n=agauss(kp_n_global,sigma_kp_n,1)
      .param vto.n=agauss(vto_n_global,sigma_vto_n,1)
      .param gamma.n=agauss(gamma_n_global,sigma_gamma_n,1)
.ends
.subckt pmod Drain Gate Source Bulk width=w length=l
      m_pmod Drain Gate Source Bulk pmod w=w l=l
      + ad='w*4u' as='w*4u' pd='2*w+8u' ps='2*w+8u'
      .MODEL pmod PMOS
      + LEVEL = 2       VTO    = vto.p   KP  = kp.p    GAMMA = gamma.p
      + PHI   = 0.72    LAMBDA = 0.011   MOB = 7       THETA = 0.154
      + NSUB  = 2E16    XJ     = 1E-6    JS  = 4.6E-3  RSH   = 115
      + TOX   = 3.6E-8  LD     = 0.75U   WD  = 0U      PB    = 0.56
      + CJ    = 3.47E-4 CJSW   = 1.73E-9 MJ  = 0.36    MJSW  = 0.39
      + AF    = 1       KF     = 2.3E-27
      .param Ab=2.984805e-8
      .param Av=5.161830e-8
      .param Ag=1.630307e-8
      .param sigma_kp_p='kp_p_global*Ab*sqrt(1/w/l)'
      .param sigma_vto_p='vto_p_global*Av*sqrt(1/w/l)'
      .param sigma_gamma_p='gamma_p_global*Ag*sqrt(1/w/l)'
      .param kp.p=agauss(kp_p_global,sigma_kp_p,1)
      .param vto.p=agauss(vto_p_global,sigma_vto_p,1)
      .param gamma.p=agauss(gamma_p_global,sigma_gamma_p,1)
.ends

*xm11 vd11 vg1 0 0 nmod w=40u l=40u
*vd11 0 vd11 -0.1
*xm12 vd12 vg1 0 0 nmod w=10u l=10u
*vd12 0 vd12 -0.1
*xm13 vd13 vg1 0 0 nmod w=5u l=5u
*vd13 0 vd13 -0.1
*xm14 vd14 vg1 0 0 nmod w=2u l=3u
*vd14 0 vd14 -0.1
*vg1 vg1 0

xm21 vd21 vg2 vs2 0 nmod w=40u l=40u
vd21 vs2 vd21 -0.1
xm22 vd22 vg2 vs2 0 nmod w=10u l=10u
vd22 vs2 vd22 -0.1
xm23 vd23 vg2 vs2 0 nmod w=5u l=5u
vd23 vs2 vd23 -0.1
xm24 vd24 vg2 vs2 0 nmod w=2u l=3u
vd24 vs2 vd24 -0.1
vg2 vg2 vs2 3.0
vs2 vs2 0

.options numdgt=6
*.dc vg1 1.5 5.0 0.035 sweep monte=56
.dc vs2 0 2 .02 sweep monte=56
*.print dc i(vd11) i(vd12) i(vd13) i(vd14)
.print dc i(vd21) i(vd22) i(vd23) i(vd24)
.end
```

Fig. 4.11: Input File for (H)Spice with Mismatch Support

Cambridge University Press, 1988.

Appendix 5: Multichip Realizations with ART 1 Modules

5.1. A Compact ART 1 Design

Saving chip area is a crucial need to improve integrated circuits yield. As yield decreases exponentially with chip area, attempting to improve yield performance has driven us to the design of a more compact and reduced prototype of the ART 1 chip.

A new prototype which implements the ART 1_m algorithm has been designed. Fig. 5.1 depicts the block diagram of the chip. As it can be seen, this new prototype diagram is equal to the one reported in Appendix 2, except for its reduced dimensions. It has an input layer of 50 input pixels, instead of 100, and it clusters the input patterns into up to 10, instead of 18, categories. However, the area reduction achieved with this prototype is greater than $100/50 \times 18/10 = 3.6$ times the area of the first prototype. The area of the actual circuitry is $6.40mm^2$ which represents a reduction of 15 times the area of the previous prototype.

This area reduction was possible thanks to the elimination of the current mirror trees which were used in the first prototype to replicate currents L_A and L_B all over the chip. These current mirror trees were used to eliminate the systematic error component in the mirror output currents [5.2]. The systematic error is due to gradients that appear in the $\beta(x,y)$, $V_{TO}(x,y)$, and $\gamma(x,y)$ surfaces (see Appendix 4). These measurements of the β, V_{TO} and γ parameters of a 6×6 transistor matrix, which occupies an area of $2.5mm \times 2.2mm = 5.5mm^2$, showed that for this order of chip dimensions the systematic error component in the transistor currents was of the same order than the random error component. The matrix of synapses in this prototype occupies an area of $2.6mm \times 0.8mm = 2.1mm^2$. Consequently, for our chip dimensions a direct replication of currents L_A and L_B using simple current mirrors with the output transistors distributed over the synapse matrix is possible without a severe output currents precision degradation.

In Appendix 4, a special purpose chip to fully characterize the parameters of a matrix of transistors was developed. For the ES2-1.0μm technology, a chip with a 6×6 cell matrix was designed, containing NMOS

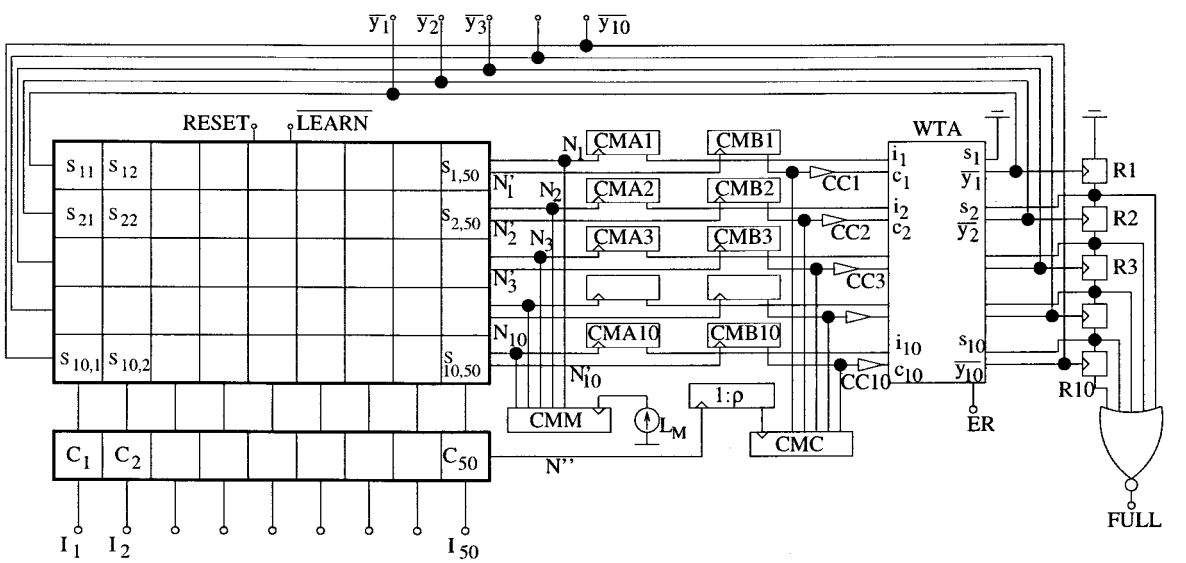


Fig. 5.1: Block diagram of new ART 1 chip prototype

and PMOS transistors of different sizes. Characteristic curves of these transistors were measured and their electrical parameters extracted. As a result, we know the parameters β, V_{TO} and γ of several arrays of transistors that have been fabricated in the ES2-1.0μm technology. If these transistors were the output transistors of a multiple-output current mirror we could predict the output currents as a function of position $I_o(x, y)$, and we could separate the contributions of the gradient and the random components.

Let us use the extracted parameters $\beta(x, y)$, $V_{TO}(x, y)$ and $\gamma(x, y)$ of a typical chip characterized in Appendix 4. For each transistor position we can compute how much each extracted parameter deviates from the mean,

$$\frac{\Delta\beta(x, y)}{\bar{\beta}} = \frac{\beta(x, y) - \bar{\beta}}{\bar{\beta}} \tag{5.1}$$

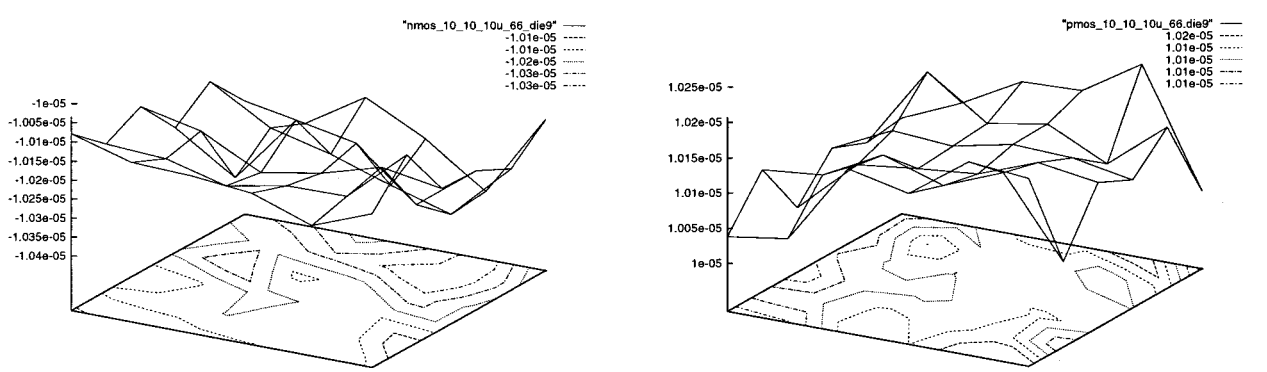
$$\Delta V_{TO}(x, y) = V_{TO}(x, y) - \overline{V_{TO}} \tag{5.2}$$

$$\Delta\gamma(x, y) = \gamma(x, y) - \bar{\gamma} \tag{5.3}$$

Using these deviations we can build a spice file with a multiple-output current mirror and obtain the surface of simulated output currents $I_o^s(x, y)$. This is shown in Fig. 5.2(a) for a NMOS multiple-output current mirror whose input current is $I_{in} = 10\mu A$ and with transistor sizes $W = L = 10\mu m$. Fig. 5.2(b) depicts the same for a PMOS multiple-output current mirror.

For each of the surfaces formed by the simulated output currents $I_o^s(x, y)$, we calculate the parameters of the plane $I_o^p(x, y) = Ax + By + C$ that best fits the points of the simulated surface $I_o^s(x, y)$. Parameters A and B represent the gradient components of the surface.

After obtaining the optimum plane $I_o^p(x, y)$ that fits the surface $I_o^s(x, y)$, we estimate the random error component present in $I_o^s(x, y)$ by computing the standard deviation of the difference $\Delta I_o(x, y) = I_o^s(x, y) - I_o^p(x, y)$. That is,



(a) **(b)**

Fig. 5.2: Typical output current distribution of a simple multiple-output current mirror as a function of output transistor position in the chip. The transistor sizes are $W = L = 10\mu m$. The input current is set to $I_{in} = 10\mu A$. (a) NMOS current mirror and, (b) PMOS current mirror

$$\sigma(\Delta I_o) = \sqrt{\frac{\sum_{n=1}^{n^{max}} \sum_{m=1}^{m^{max}} \left(I_o^s(n\Delta x, m\Delta y) - I_o^p(n\Delta x, m\Delta y) - \overline{\Delta I_o} \right)^2}{n^{max} \times m^{max}}} \tag{5.4}$$

where n^{max} and m^{max} are the number of times each transistor is repeated in the x and y directions and, Δx and Δy are the distances between two adjacent transistors in the x and y directions, respectively.

The total standard deviation (random+systematic) can also be computed as,

$$\sigma_T(I_o) = \sqrt{\frac{\sum_{n=1}^{n^{max}} \sum_{m=1}^{m^{max}} \left(I_o^s(n\Delta x, m\Delta y) - \overline{I_o^s} \right)^2}{n^{max} \times m^{max}}}. \tag{5.5}$$

Simulations were performed using the extracted parameters of the 6×6 matrixes of the NMOS transistors of size $W = L = 10\mu m$ located in 8 different chips. The input current level was set to $10\mu A$.

Let us call the maximum value of the interpolated plane $I_{maxplane}$, that is,

$$I_{maxplane} = max\{ I_o^p(x, y) \} \tag{5.6}$$

and $I_{minplane}$ the minimum value

$$I_{minplane} = min\{ I_o^p(x, y) \}. \tag{5.7}$$

Let us compute the maximum systematic deviation in the output currents as

$$\Delta I_o^p = I_{maxplane} - I_{minplane} \tag{5.8}$$

Since 98% of the random values remain within the interval $\pm 3\sigma(\Delta I_o)$, let us measure the relationship between the random error component and the systematic error component by the following ratio

$$r^{ran/sys} = \frac{6 \times \sigma(\Delta I_o)}{\Delta I_o^p} \tag{5.9}$$

Table 5.1 contains the values of the random deviation component $\sigma(\Delta I_o)$, the gradient components A and B, the systematic deviation ΔI_o^p, the relation $r^{ran/sys}$, and the total standard deviation $\sigma_T(I_o)$ obtained for the 8 different chips. These results are for chips of size similar to $2.5mm \times 2.2mm = 5.5mm^2$. As it can be observed in Table 5.1, for this chip sizes the systematic error component is of the same order (and usually less) than the random deviation component.

The random error component is quite stable from chip to chip and its mean value averaged over the 8 chips is $\overline{\sigma(\Delta I_o)} = 0.059\mu A$, which means a relative random error in the output currents of approximately 0.59%. The total error component averaged over all the chips is $\overline{\sigma_T(I_o)} = 0.076\mu A$, which is a total relative error of 0.76%.

Similar simulations were performed for 6×6 matrixes of PMOS transistors of size $W = L = 10\mu m$ for the 8 different chips. The input current was again set to $10\mu A$. Table 5.1 contains the values of $\sigma(\Delta I_o)$, A, B, ΔI_o^p, $r_{ran/sys}$, and $\sigma_T(I_o)$ obtained for each chip.

The random error component is again quite stable from chip to chip, with a mean value averaged over the 8 chips of $\overline{\sigma(\Delta I_o)} = 0.046\mu A$, which is equivalent to a relative random current error of 0.46%. The systematic error component has again large variations from chip to chip but remains in the same order of magnitude than the random error component. The total standard deviation averaged over all the chips is $\overline{\sigma_T(I_o)} = 0.057\mu A$, which is a total relative error of 0.57%.

Based on these results we designed another ART 1 prototype that contains a matrix of 50×10 synapses. The synapse matrix occupies an area of $2.6mm \times 0.8mm = 2.1mm^2$. Each synapse contains two L_A current sources and one L_B current source. Fig. 5.3 contains a detailed schematic of each synapse circuit. In this prototype, each L_A current source is an output transistor of a simple NMOS current mirror of size $W = L = 10\mu m$. Similarly, each L_B current source is an output transistor of a simple PMOS current mirror of size $W = L = 10\mu m$.

chip	$\sigma(\Delta I_o)$ (μA)	A (μA)	B (μA)	ΔI_o^p (μA)	$r^{ran/sys}$	$\sigma_T(I_o)$ (μA)
1	0.057368	0.000990	-0.020644	0.129800	2.652	0.067357
2	0.061811	-0.032629	0.000362	0.197943	1.874	0.083224
3	0.047458	0.029508	0.022027	0.309206	0.921	0.078784
4	0.051876	-0.011395	-0.003614	0.090057	3.456	0.055749
5	0.053839	-0.009419	-0.018067	0.164915	1.959	0.064105
6	0.058233	0.035253	0.014964	0.301303	1.160	0.087573
7	0.065038	-0.003324	-0.029267	0.195543	1.996	0.082222
8	0.072544	-0.031138	0.004652	0.214743	2.027	0.090298

Table 5.1: Output current error in a 6×6 NMOS current mirror

chip	$\sigma(\Delta I_o)$ (μA)	A (μA)	B (μA)	ΔI_o^p (μA)	$r^{ran/sys}$	$\sigma_T(I_o)$ (μA)
1	0.058207	-0.017879	-0.007673	0.153314	2.278	0.067023
2	0.047461	-0.010970	-0.001420	0.074343	3.830	0.051083
3	0.048464	-0.006271	0.007500	0.082628	3.519	0.051260
4	0.039949	0.026079	0.010238	0.217897	1.100	0.063231
5	0.046463	-0.008931	0.001028	0.059749	4.666	0.048933
6	0.045003	0.032343	-0.004070	0.218475	1.236	0.071586
7	0.044067	0.013397	-0.000502	0.083394	3.171	0.049660
8	0.041021	-0.015243	-0.006057	0.127800	1.926	0.049673

Table 5.2: Output current error in 6×6 PMOS current mirrors

There are two analog switches controlled by signal s_i that disconnect the output currents of the synapse from nodes N_j and N_j'. These signals s_i, which are common to all the synapses in the same column, are generated by a column selection decoder. For each combination of the decoder input signals, only one column of synapses is injecting its output currents into nodes N_j and N_j'. During the normal circuit operation, an enable input signal to the decoder is activated that allows all the signals s_i to be high at the same time. Thus, all the synapses inject their output currents into nodes N_j and N_j'.

Using a computer which controlled the column selection decoder and a DC curve tracer also controlled by the computer, we were able to measure separately the output current flowing through each of the current sources L_{A1}^{ij}, L_{A2}^{ij} and L_B^{ij} in each synapse. Before doing these measurements, all the weights z_{ij} were reset to their high state and all the input vector components I_i were loaded with '1' through the shift register. Currents L_{A1}^{ij} and L_{A2}^{ij} of each synapse were measured setting the input current L_B to 0. Then, using the selection circuitry we alternatively connected a different column to nodes N_j and N_j' measuring the current flowing into these nodes. Afterwards, current L_A was set to 0 and a current was injected through the input of the L_B current mirror. The output current L_B^{ij} of each synapse flowing into nodes N_j was measured.

Fig. 5.4(a) shows the measured output currents of the synapse current sources L_{A1}^{ij} in one chip. The input current L_A was set to $10\mu A$. To obtain the surface shown in Fig. 5.4, we have represented the L_{A1}^{ij} current as a function of the coordinates (x, y) of the synapse where the source is located. Fig. 5.4(b) depicts the output currents L_{A2}^{ij} of the same chip for the same input current level $L_A = 10\mu A$. Fig. 5.4(c) shows the surface obtained when we represent the synapse output currents L_B^{ij} of a chip when the input current L_B is set to $10\mu A$.

To compute the random error component and the systematic deviation component of these surfaces we follow the same procedure used with the simulated surfaces. For each of the L_{A1}, L_{A2} and L_B surfaces, we calculate the A, B and C parameters of the plane $L_K^p(x, y) = Ax + By + C$ that best fits the measured surface $L_K^m(x, y)$. Afterwards, we compute the standard deviation of the difference $\Delta L_K(x, y) = L_K^m(x, y) - L_K^p(x, y)$, that is

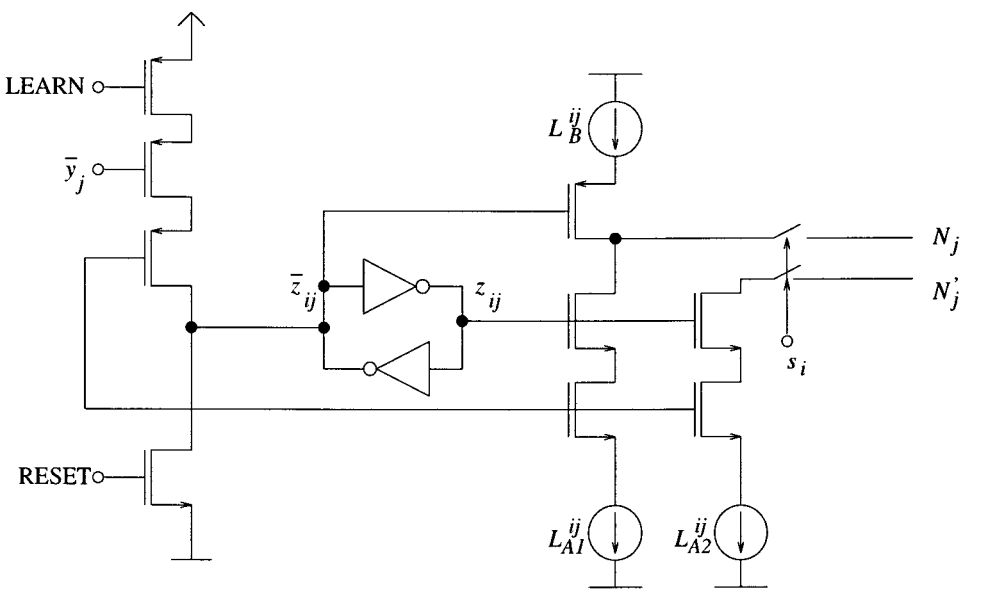
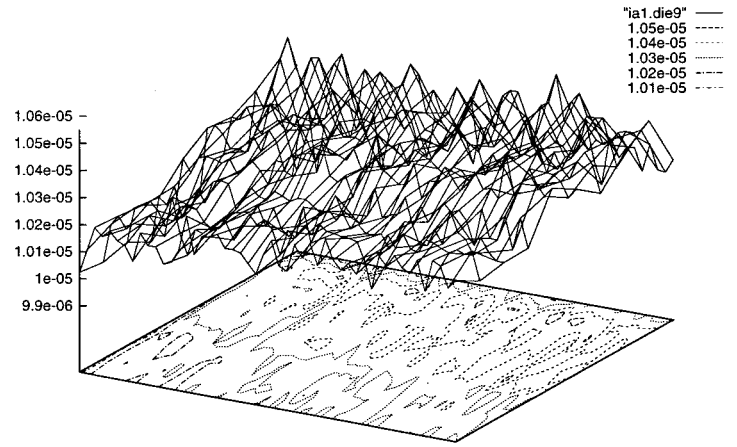
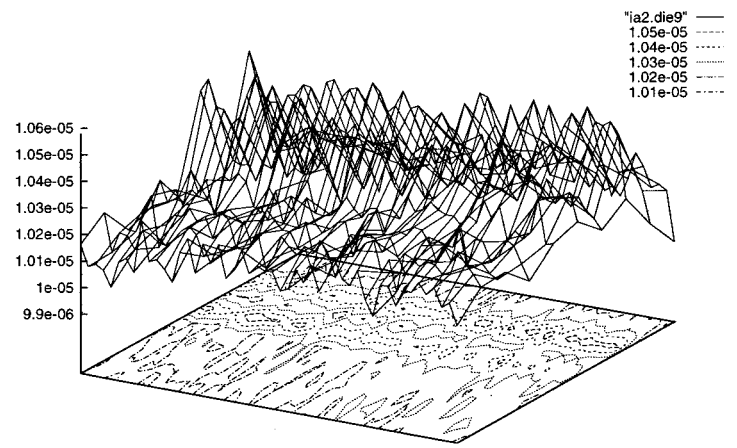


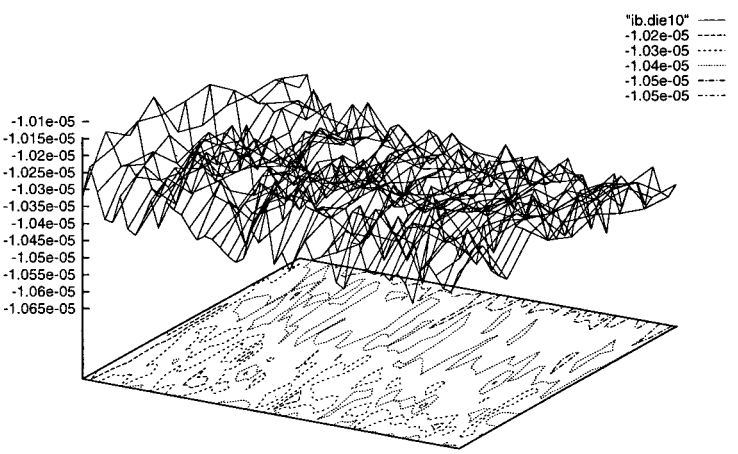
Fig. 5.3: Diagram of a synapse in the ART 1 chip



(a)



(b)



(c)

Fig. 5.4: Representation of the synapse output current as a function of the synapse position in the chip, (a) L_{A1} output currents, (b) L_{A2} output currents and, (c) L_B output currents

$$\sigma(\Delta L_K) = \sqrt{\frac{\sum_{n=1}^{n^{max}} \sum_{m=1}^{m^{max}} \left(L_K^m(n\Delta x, m\Delta y) - L_K^p(n\Delta x, m\Delta y) - \overline{\Delta L_K} \right)^2}{n^{max} \times m^{max}}}, \tag{5.10}$$

which is a measurement of the random error component in the output current L_K.

We also find the maximum value of the output current obtained for a chip $L_{K_{maxplane}}$, that is,

$$L_{K_{maxplane}} = max\{L_K^p(x,y)\} \tag{5.11}$$

the minimum output current $L_{K_{minplane}}$ for a chip

$$L_{K_{minplane}} = min\{L_K^p(x,y)\}, \tag{5.12}$$

the maximum systematic deviation in the output currents

$$\Delta L_K^p = L_{K_{maxplane}} - L_{K_{minplane}} \tag{5.13}$$

and the relation between the random error component and the systematic error component

$$r_K^{ran/sys} = \frac{6 \times \sigma(\Delta L_K)}{\Delta L_K^p} \tag{5.14}$$

Table 5.3 contains, for each chip, the values computed for the random error component $\sigma(\Delta L_{A1})$, the gradient components A and B, the systematic deviation component ΔL_{A1}^p, the relation between the random and systematic deviation $r_{A1}^{ran/sys}$, and the total standard deviation in the output currents $\sigma_T(L_{A1})$.

The mean value of the random error component averaged over all the chips $\overline{\sigma(\Delta L_{A1})}$ is $0.065\mu A$, which is equivalent to a relative error of 0.65%. The mean value of the total standard deviation $\overline{\sigma_T(L_{A1})}$ is

chip	$\sigma(\Delta L_{A1})(\mu A)$	$A\ (nA)$	$B\ (nA)$	$\Delta L_{A1}^p\ (\mu A)$	$r_{L_{A1}}^{ran/sys}$	$\sigma_T(L_{A1})(\mu A)$
1	0.062479	0.711	10.361	0.139149	2.694	0.070785
2	0.050822	0.233	5.588	0.067536	5.311	0.062175
3	0.067802	2.660	-5.821	0.191207	2.128	0.080585
4	0.059039	0.544	0.089	0.028099	12.607	0.061145
5	0.063774	-1.179	6.046	0.126529	3.204	0.069474
6	0.064865	-1.981	2.947	0.128540	3.028	0.071346
7	0.065738	-0.125	-3.512	0.041365	9.535	0.068456
8	0.064164	-0.773	5.358	0.092227	4.174	0.066945
9	0.079046	2.258	10.701	0.219908	2.157	0.090626
10	0.074479	-0.529	1.666	0.043103	10.368	0.075308

Table 5.3: Measured output current error in the L_{A1} NMOS simple current mirror

$0.072\mu A$, which is a relative error in the output currents of 0.72%. Consequently, for these dimensions the gradient component does not cause a serious degradation in the current deviations.

Table 5.3 contains the same information than Table 5.3 but for the L_{A2} output currents. The random standard deviation averaged over all the chips is $0.073\mu A$, that is, a 0.73% of relative error. The mean value of the total standard deviation is $0.078\mu A$, a 0.78% relative error.

Table 5.3 gives the measured deviation in the synapse current sources L_B. The mean of the random relative error is 0.62% and the mean of the total relative error is 0.70%.

5.2. Experimental Results of this ART 1 Prototype

As mentioned in the previous section, we have designed and fabricated a new prototype with an F_1 layer of $N = 50$ input pixels and a category layer with $M = 10$ categories. The chip has been designed and

chip	$\sigma(\Delta L_{A2})(\mu A)$	$A\ (nA)$	$B\ (nA)$	$\Delta L_{A2}^{p}\ (\mu A)$	$r_{L_{A2}}^{ran/sys}$	$\sigma_T(L_{A2})\ (\mu A)$
1	0.069025	0.796	9.600	0.135811	3.057	0.075334
2	0.063631	-0.198	5.028	0.059737	6.391	0.064838
3	0.067669	2.419	-7.264	0.193584	2.097	0.080359
4	0.067981	0.207	-0.475	0.015092	27.027	0.070033
5	0.064327	-1.589	4.707	0.126529	3.050	0.070788
6	0.071239	-2.190	1.679	0.126294	3.384	0.078401
7	0.064670	-0.469	-1.609	0.039520	9.818	0.066243
8	0.063205	-1.179	6.548	0.124456	3.047	0.069372
9	0.107896	1.157	7.259	0.130448	4.962	0.112106
10	0.088077	-0.802	-1.394	0.054058	9.776	0.090044

Table 5.4: Measured output current error in the L_{A2} NMOS simple current mirror

chip	$\sigma(\Delta L_B)\ (\mu A)$	$A\ (nA)$	$B\ (nA)$	$\Delta L_B^{p}\ (\mu A)$	$r_{L_B}^{ran/sys}$	$\sigma_T(L_B)\ (\mu A)$
1	0.062361	0.914	-1.588	0.061606	6.076	0.063976
2	0.059196	-0.126	-1.524	0.021530	16.497	0.060367
3	0.056222	3.203	-17.212	0.332290	1.015	0.089202
4	0.062793	-1.043	-3.764	0.089795	4.196	0.064319
5	0.064517	-2.690	-4.824	0.182743	2.118	0.075611
6	0.063882	-2.642	-1.728	0.149403	2.565	0.073131
7	0.059516	-1.182	-9.926	0.158375	2.255	0.066926
8	0.062204	-2.424	-2.666	0.147884	2.524	0.070645
9	0.062816	-0.059	-3.444	0.037389	10.080	0.063074
10	0.056737	-1.315	-15.073	0.216458	1.573	0.072737

Table 5.5: Measured output current error in the L_B PMOS current mirror

fabricated in the double-metal single-poly CMOS technology of ES2-1.0μm, and has been mounted on a PGA-84 pins package. The circuit area is $6.40mm^2$, but for our test prototype the total area is $18.82mm^2$ which is due to the pads. An area reduction of around 15 times has been achieved in comparison with the prototype described in Appendix 2.

This area reduction resulted in a great yield improvement. In this case, we obtained a 100% yield compared to the poor 6.7% yield of the first prototype. In Appendix 2, using the results of the yield of our first prototype, and considering a dependence between the yield performance, the die area Ω, and the process defect density ρ_D, given by the expression,

$$yield\,(\%) \;=\; 100e^{-\rho_D\Omega} \tag{5.15}$$

we estimated a process defects density ρ_D of $3.2cm^{-1}$. If we use this value of ρ_D to estimate the yield of a chip of area $\Omega = 0.064cm^2$ a yield performance of 98% results. Therefore, the yield improvement achieved now is totally justified by the area reduction.

The system level operation of this prototype has been tested not only for a single chip, but also when several chips were assembled horizontally to increase the size of the input patterns. In the next subsections, we show the test results for a single chip, and for a system formed by two horizontally assembled chips.

A. Single ART 1 Chip Operation

The operation of a single chip has been tested using the digital test equipment HP82000. This equipment automatically applied a sequence of binary input patterns, and also read the winning category and the stored weights after the classification and learning of each input pattern has taken place.

To test the system we have trained the system with a set of ten $7 \times 7 = 49$ input patterns. Each pattern represents each of the ten digits from '0' to '9'. The last input pixel was always set to zero and it is not shown in the figures. The classification of the set of input patterns was repeated for different values of the vigilance parameter ρ and several values of parameter $\alpha = L_A/L_B$.

Fig. 5.5 shows the training sequence for a vigilance $\rho = 0.3$ and $\alpha = 1.1$. The first column represents the input pattern applied to the system. The remaining ten columns correspond to the weights stored in each category when the input pattern has been classified and learned. The underlined category is the winning category after the Winner-Take-All competition. In this case, learning self-stabilizes after two input pattern presentations. That is, no modification of the winning category or the stored weights take place in subsequent presentations of the input patterns. As shown if Fig. 5.5, the system has classified the ten input patterns into four categories.

Fig. 5.6 shows the complete training sequence obtained for the same value of $\alpha = 1.1$ but for a higher vigilance parameter, $\rho = 0.5$. Now, the system needs only one iteration of the input pattern set until learning self-stabilizes. Due to the higher vigilance parameter, the system forms more categories to classify the same input patterns. The system classifies the ten input patterns into six different categories. A similar effect occurs, as explained in Appendix 1, when the vigilance parameter remains constant but we increase the current ratio

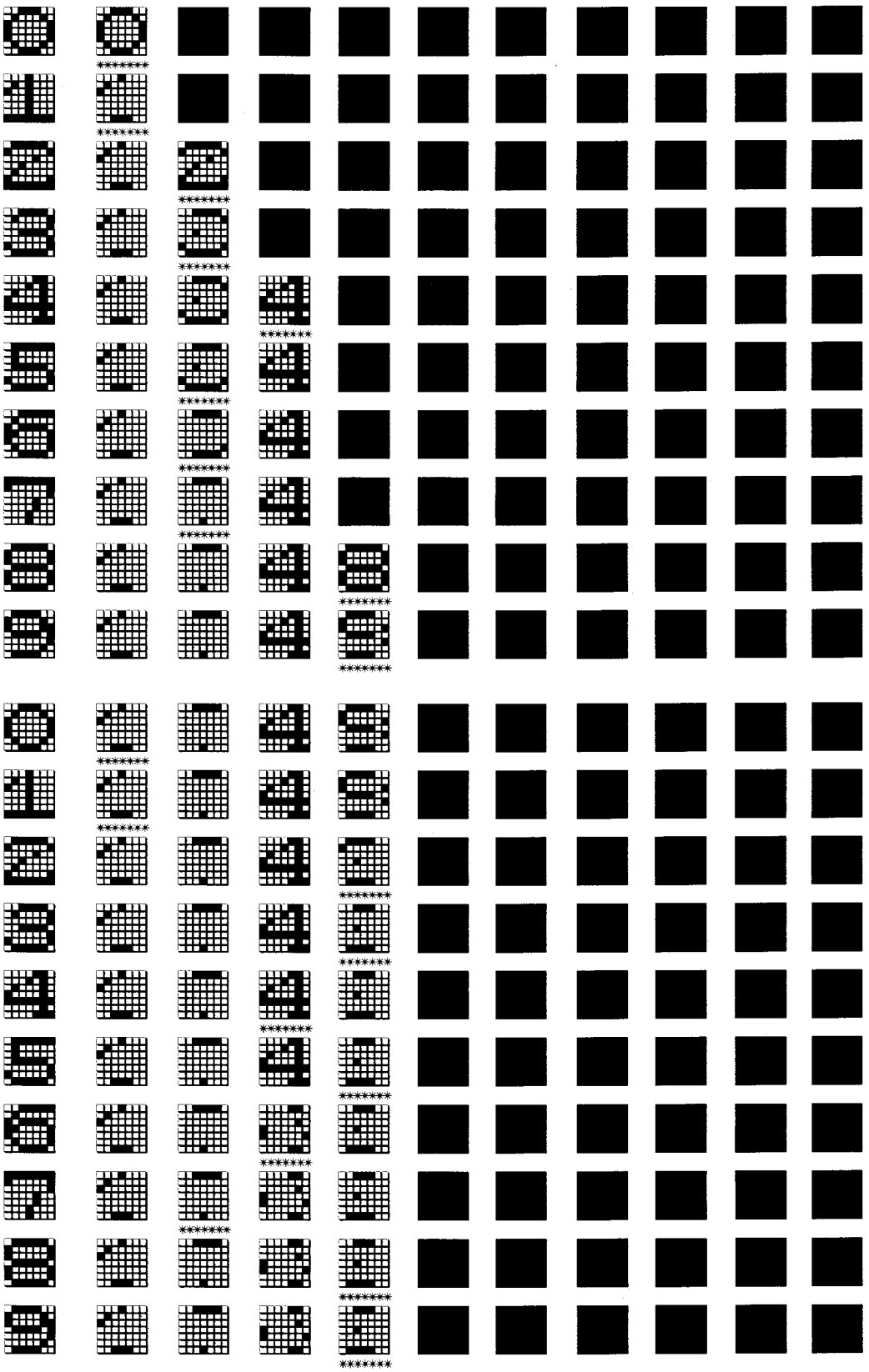


Fig. 5.5: Classification performed by a chip for $\rho = 0.3$ and $\alpha = 1.1$

α. Fig. 5.7 shows the complete training sequence performed by the chip when the vigilance parameter is $\rho = 0.3$ but the current ratio has been increased to $\alpha = 3.11$. After three iterations the system forms six categories to classify the ten input patterns, as occurred when $\rho = 0.5$ and $\alpha = 1.1$

B. Multichip ART 1 Operation

A system composed of two horizontally connected chips was arranged. This system can cope with input patterns of $2 \times N$ binary pixels. Fig. 5.8 shows a diagram of two horizontally interconnected chips. To expand the system in an horizontal way, nodes N_j, N_j', and N'' of the different chips have to be interconnected and isolated, all except one of them, from the *CMA'S* and *CMB'S* current mirrors, and the adjustable gain

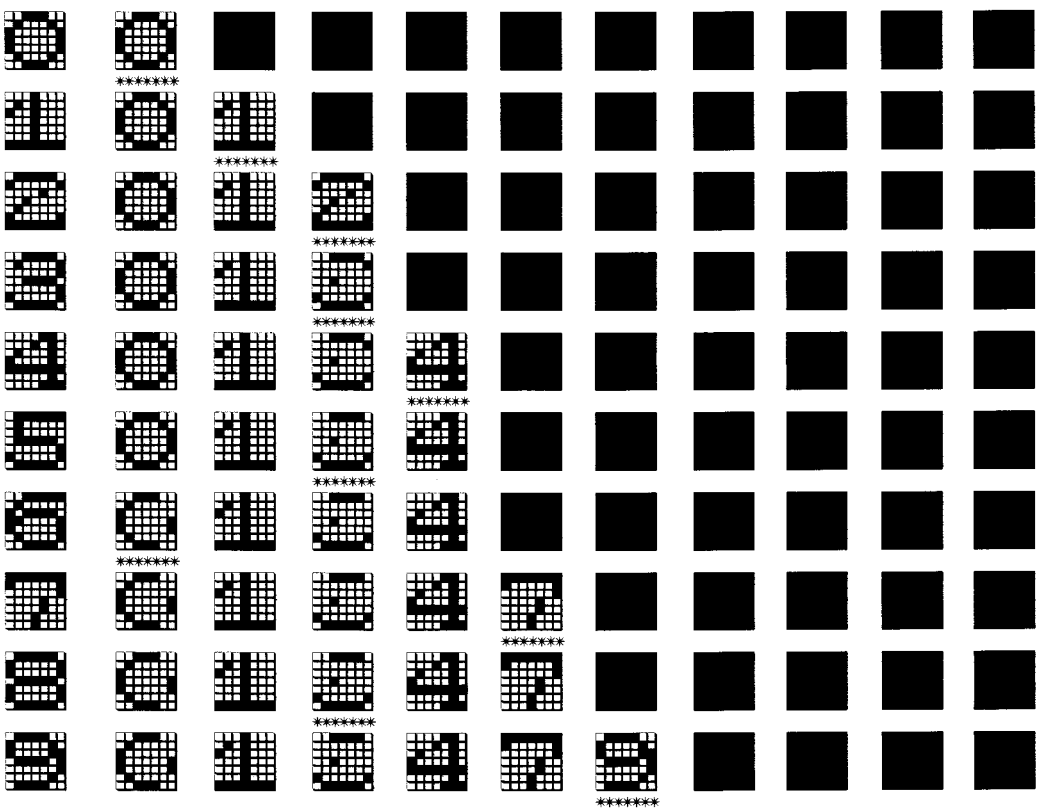


Fig. 5.6: Classification performed by a single chip with a vigilance parameter $\rho = 0.5$, and $\alpha = 1.1$

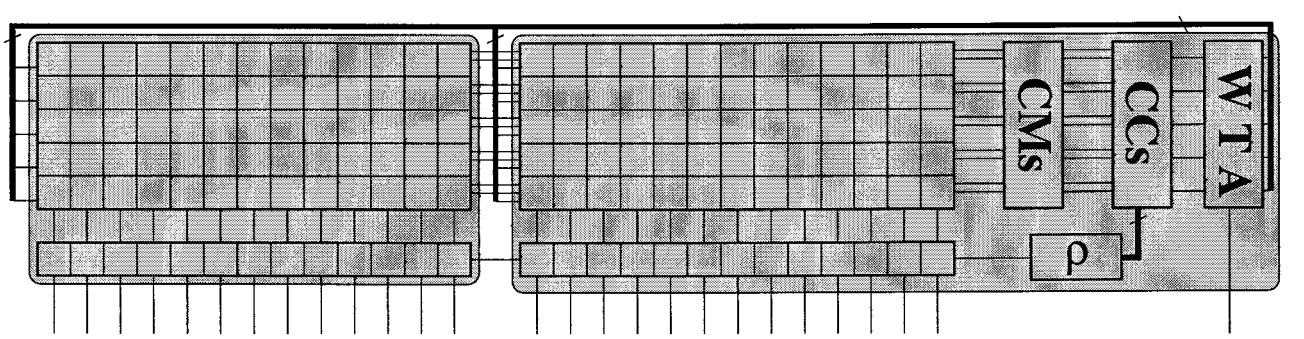


Fig. 5.8: Interconnection of two chips for horizontal system expansion

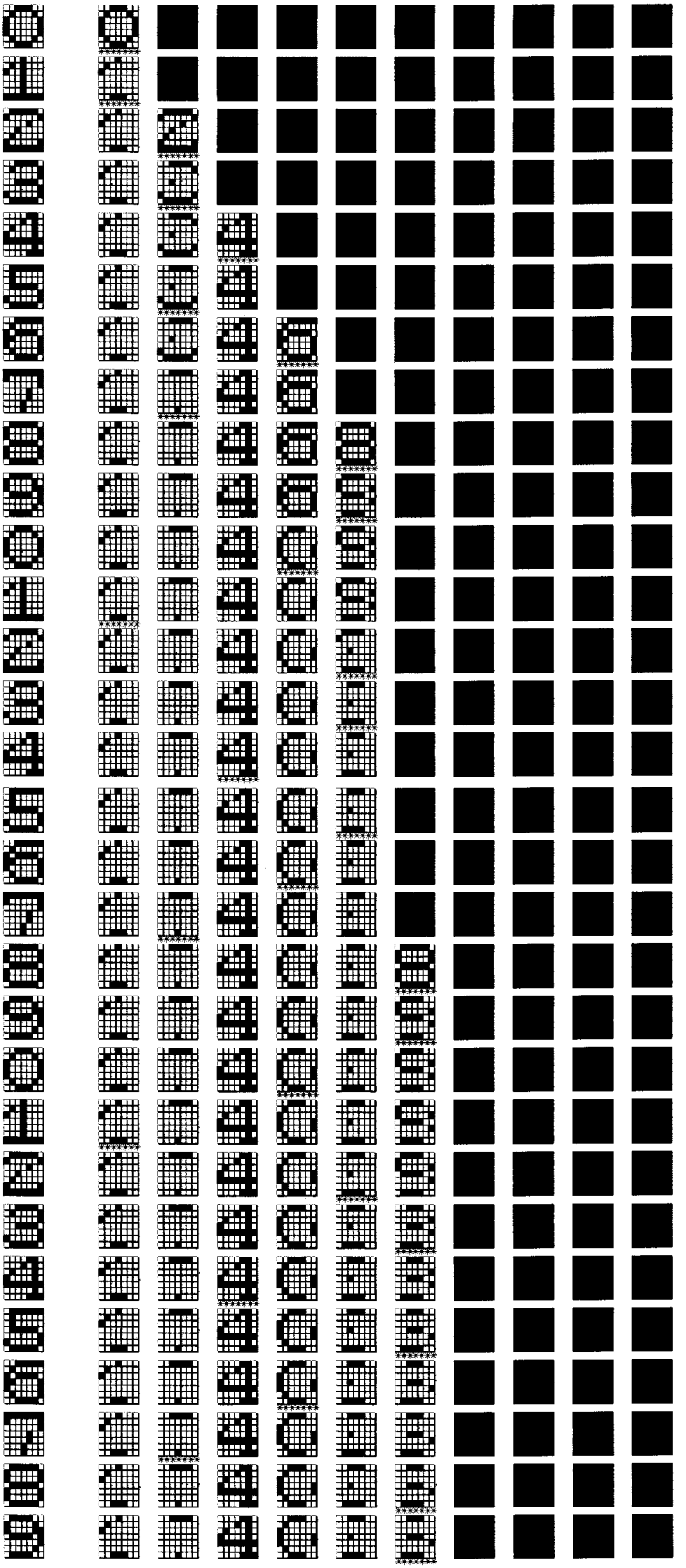


Fig. 5.7: Training sequence of a chip for $\rho = 0.3$ and $\alpha = 3.11$

ρ-mirror. The outputs y_j of the active *WTA* have also to be shared among all the chips to control the weights updating in all synapses.

The system level performance of the two assembled chips has been tested. In this case, the input patterns had $10 \times 10 = 100$ binary pixels. Fig. 5.9 depicts a training sequence performed on the system. The system classifies the 10 input patterns into 8 categories after a single presentation of the set of input patterns. The sequence of Fig. 5.9 was obtained with the vigilance parameter set to $\rho = 0.5$, and the current levels in the synapses $L_A = 10\mu A$ and $L_B = 5\mu A$, that is, $\alpha = 2$.

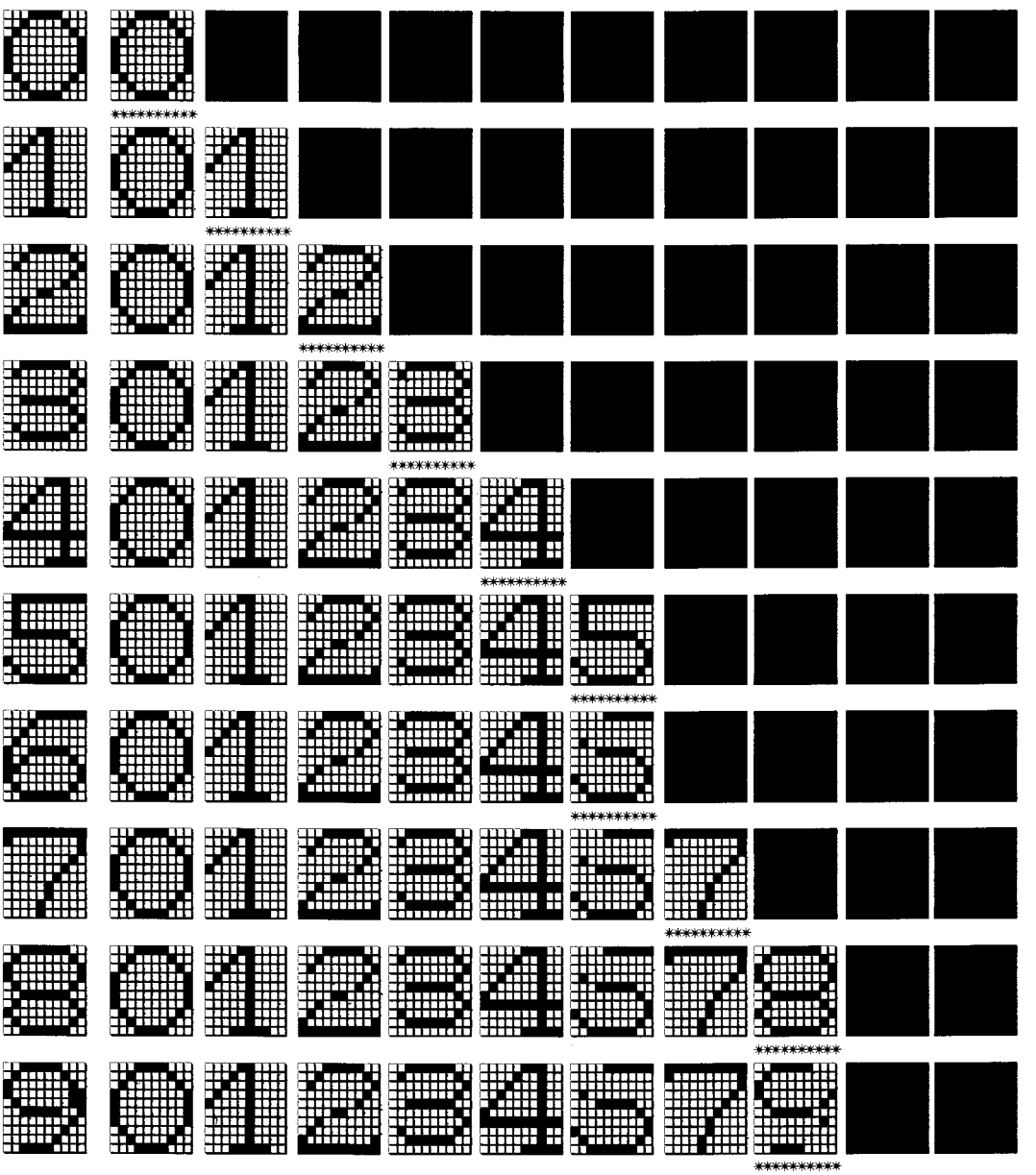


Fig. 5.9: Training sequence of a system formed by two horizontally assembled chips. The vigilance parameter is $\rho = 0.5$ and $\alpha = 2$

5.3. ARTMAP Architectures

An ARTMAP architecture is a system which can be trained in a supervised way to learn the correspondence between pairs of binary input patterns [5.3].

A. The ARTMAP Algorithm: Supervised Learning

Fig. 5.10 shows a block diagram of the ARTMAP architecture. It consists of two ART 1 modules ($ART\ 1^a$ and $ART\ 1^b$) and an inter-ART module. We denote as $\boldsymbol{a} = (a_1, a_2, ..., a_{N_a})$ the N_a-dimensional input vector to module $ART\ 1^a$, and $\boldsymbol{b} = (b_1, b_2, ..., b_{N_b})$ the N_b-dimensional input vector to $ART\ 1^b$. Whenever a pair of input vectors, **a** and **b**, is presented, self-organization processes evolve in the $ART\ 1^a$ and $ART\ 1^b$ modules. Then, the inter-ART module learns to recognize the correspondence between the categories activated in $ART\ 1^a$ and $ART\ 1^b$. The "match-tracking" signal is intended to control the self-organization process when an error prediction takes place. If the activated category in $ART\ 1^a$ has previously learned to predict an $ART\ 1^b$ category which is different than the activated one, the inter-ART module sends a match-tracking signal to the $ART\ 1^a$ module. This match-tracking signal influences the $ART\ 1^a$ vigilance parameter ρ_a. It increases this vigilance parameter by the minimum amount necessary to force the $ART\ 1^a$ system to reset the currently activated category. This process of $ART\ 1^a$ searches and match-tracking reset continues until the $ART\ 1^a$ system chooses an F_2^a category which predicts the activated category in F_2^b or has not previously learned any other F_2^b prediction.

Fig. 5.11 depicts a more detailed diagram of the ARTMAP architecture. In this figure, letter a denotes the elements of the $ART\ 1^a$ module. Letter b denotes the elements associated with $ART\ 1^b$. As we can see in Fig. 5.11 there is a one to one correspondence between the nodes of the $ART\ 1^b$ category layer and the nodes in the "mapfield" module F^{ab}. Each F_2^b node is connected to an F^{ab} node bidirectionally through a vector of

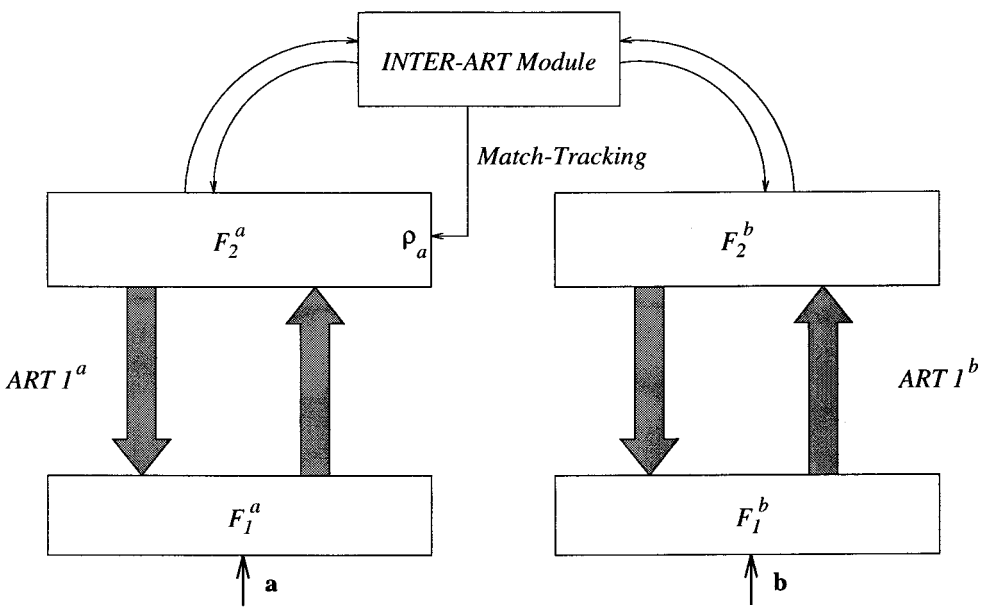


Fig. 5.10: ARTMAP Architecture

non-adaptive weights which always equal '1'. However, the nodes of the $ART\ 1^a$ category layer are interconnected to the mapfield ones by a matrix of $M_a \times M_b$ adaptive binary weights $\{w_{jk}\}$. We have denoted as $\mathbf{y}^{ab}$ the activation pattern across the nodes of F^{ab}. As shown in Fig. 5.11, each node u_k^{ab} in the map-field layer receives inputs from three sources of activation: the control signal G, the F_2^b output signals y_k^b, and the gated F_1^a output signals $\sum_j w_{jk} y_j^a$. The activation of a u_k^{ab} node, obeys the 2/3 rule: signal y_k^{ab} is activated if at least 2 of the 3 node inputs are high. Mathematically,

$$y_k^{ab} = \begin{cases} 1 \text{ if } y_k^b + G + \sum_{j=1}^{M_a} y_j^a w_{jk} \geq 2 \\ 0 \quad \text{otherwise} \end{cases} \tag{5.16}$$

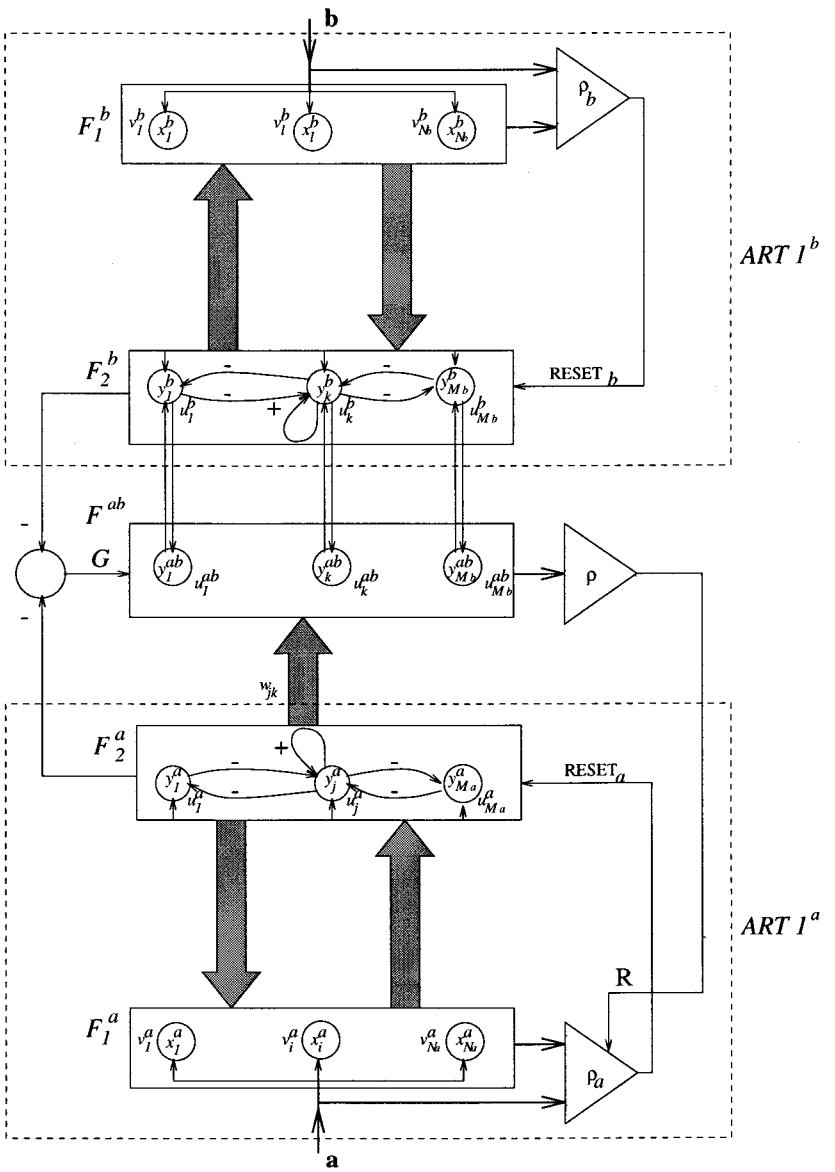


Fig. 5.11: Diagram of the interactions in the ARTMAP architecture

where G is a control signal which is always active except when the F_2^a and F_2^b layers are simultaneously active. That is,

$$G = \begin{cases} 0 \text{ if } F_2^a \text{ and } F_2^b \text{ are active} \\ 1 \text{ otherwise} \end{cases} \tag{5.17}$$

The function of signal G is to allow the system to distinguish between two operating modes: the training mode and the prediction mode.

In the training mode two input vectors, **a** and **b**, are applied to the system which must learn the correspondence between the activated F_2^a and F_2^b categories. In the prediction mode, only one input pattern **a** is presented to the $ART\ 1^a$ module and the system must predict the corresponding F_2^b category.

The adaptive weights w_{jk} are binary valued ones. They are initially set to '1' and during the system training follow the learning rule

$$w_{jk}^{new} = \begin{cases} y_k^{ab} \text{ if } y_j^a = 1 \\ w_{jk}^{old} \text{ if } y_j^a = 0 \end{cases} . \tag{5.18}$$

According to this law, the weight vector $\boldsymbol{w}_J$ corresponding to the active F_2^a node v_J^a, evolves towards the activation pattern across F^{ab}, $\boldsymbol{y}^{ab}$. The weight vectors connecting the remaining F_2^a nodes, $\boldsymbol{w}_j$ with $j \neq J$ remain constant.

As mentioned above, the system can operate in two different modes:

- Training mode: Two input patterns, **a** and **b**, are applied to layers F_1^a and F_2^b, respectively. The $ART\ 1^a$ and $ART\ 1^b$ modules classify the input patterns in their corresponding categories, u_J^a and u_K^b. The presence of simultaneous activation in layers F_2^a and F_2^b will deactivate the control signal $G = 0$. The activation vector across layer F^{ab} will become,

$$y_k^{ab} = \begin{cases} 1 \text{ if } k=K \text{ and } w_{JK}=1 \\ 0 \text{ otherwise} \end{cases} \tag{5.19}$$

  which can be expressed as,

$$y_k^{ab} = y_k^b w_{Jk} \tag{5.20}$$

  or in vector notation,

$$\boldsymbol{y}^{ab} = \boldsymbol{y}^b \cap \boldsymbol{w}_J . \tag{5.21}$$

  This means that all nodes u_k^{ab} with $k \neq K$ will be deactivated, node u_K^{ab} will be active only if node u_J^a has previously learned to predict category u_K^b, or if node u_J^a has not previously learned to predict any other $ART\ 1^b$ category.

If node u_J^a has previously learned to predict an F_2^b category different from the activated one u_K^b, no node in F^{ab} will be active. In this case, the match-tracking signal R will become active. Signal R increases the $ART\ 1^a$ vigilance parameter ρ_a by the minimum amount necessary to deactivate the node u_J^a which has caused the prediction error. Therefore, the new vigilance parameter $\overline{\rho_a}$ will satisfy the following condition,

$$\overline{\rho_a} > \frac{|\boldsymbol{a} \cap z_J^a|}{|\boldsymbol{a}|}. \tag{5.22}$$

The search process in $ART\ 1^a$ continues, increasing each time the vigilance parameter ρ_a, until a category u_J^a is found which has previously learned to predict category u_K^b, or which has not predicted any $ART\ 1^b$ category yet.

After the search process is completed, the weights are updated. Weights z_{ij}^a and z_{kl}^b follow the ART 1 algorithm learning rules, while weights w_{jk} follow the learning rule given by (5.18).

- Prediction mode: In this operation mode only an input pattern **a** is presented to the $ART\ 1^a$ module and the system must predict the corresponding $ART\ 1^b$ category.

  Pattern **a** will activate an u_J^a category in F_2^a, while F_2^b remains inactive. In this case, the control signal G will be active. The activation pattern across the inter-ART module F^{ab} will become,

$$y_k^{ab} = \begin{cases} 1 \text{ if } w_{Jk}=1 \\ 0 \text{ otherwise} \end{cases}. \tag{5.23}$$

  Category u_J^a activates all the nodes in F^{ab} if u_J^a has not previously learned any prediction. If a correspondence between u_J^a and a node u_K^b has been previously learned, only the corresponding node u_K^{ab} will be active because w_{JK} is the only weight that will have a '1' value.

Fig. 5.12 depicts the flow diagram of the ARTMAP operation in the training mode case and Fig. 5.13 shows the ARTMAP flow diagram in the prediction mode operation. In both diagrams, the operation of the $ART\ 1^a$ and $ART\ 1^b$ modules is considered to be that of the ART 1_m algorithm reported in Appendix 1.

B. ARTMAP Circuit Implementation

A circuit that implements the ARTMAP algorithm has been realized. To construct the ARTMAP system, we have interconnected two chips which implement the ART 1_m algorithm through an inter-ART module. Fig. 5.14 shows the interconnection diagram of the ARTMAP system and Fig. 5.15 shows the detailed schematic of the inter-ART circuit.

The inter-ART circuit has been designed and fabricated using the same technology as the ART 1 chip: the ES2-1.0μm double-metal single-poly CMOS technology. It consists of a matrix of $M_a \times M_b$ (in our case, it is a 10×10 matrix) cells that occupies an area of $0.459mm \times 0.478mm = 0.219mm^2$ and it is mounted on a DIL-28 pins package.

Each cell c_{jk} contains a memory element which stores the value of the weight w_{jk} plus some circuitry to reset and update the weights. All the cells in the same column j share the input y_j^a, which is the output from node u_j^a in F_2^a. In the same way, the outputs y_k^b of layer F_2^b are shared as inputs by all the inter-ART cells in the same row k. The reset signal "RESET", t and the "LEARN" signal are shared by all the cells c_{jk} in the inter-ART chip. Signals "RESET" and "LEARN" are also common to the three chips in the system. There is also some circuitry in each cell to read out the value of the weight w_{JK}, corresponding to the activated cells u_J^a and u_K^b.

The system performs the operation sequence depicted in Fig. 5.12 and Fig. 5.13:

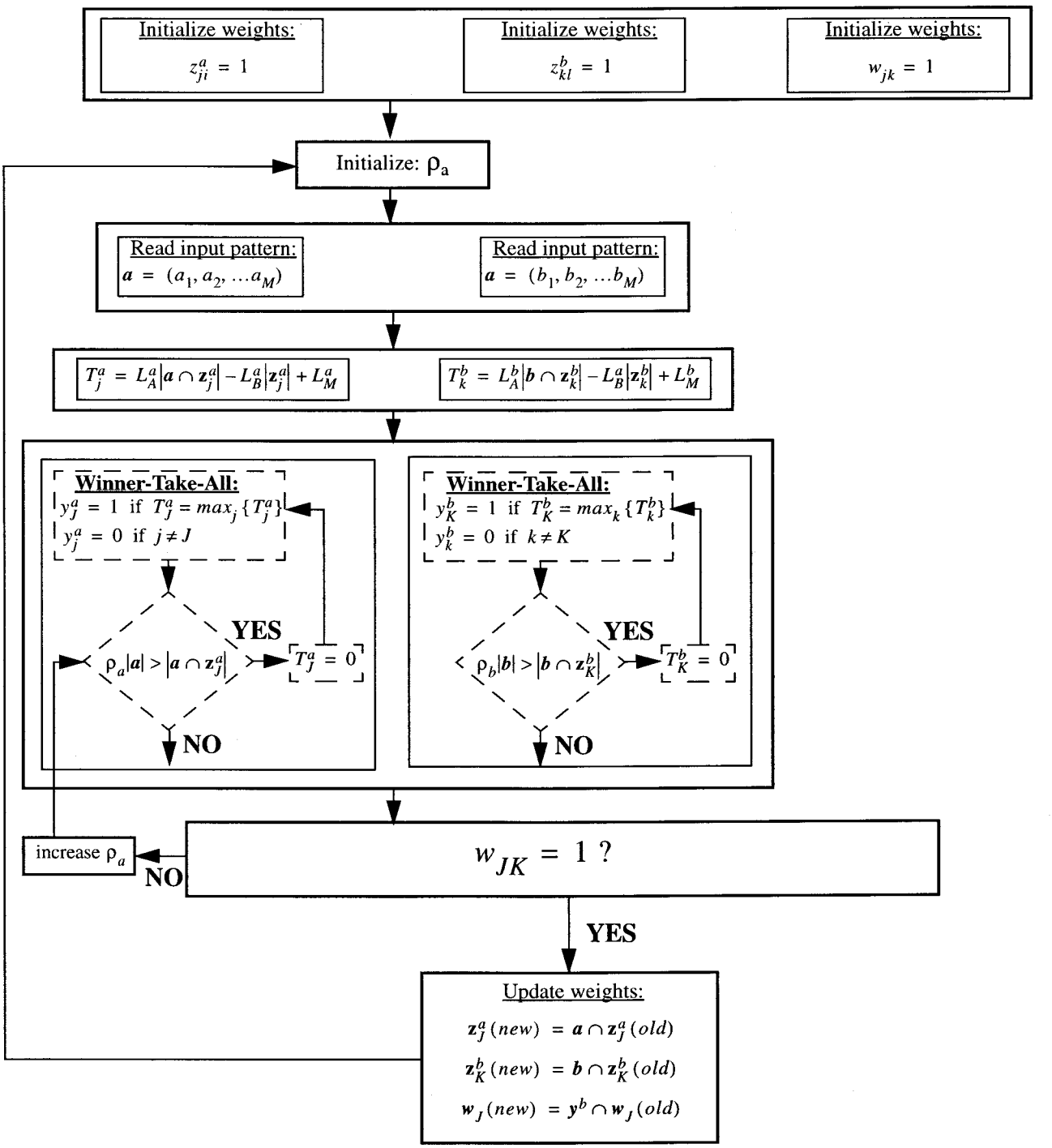


Fig. 5.12: Flow diagram of the ARTMAP operation during the training phase

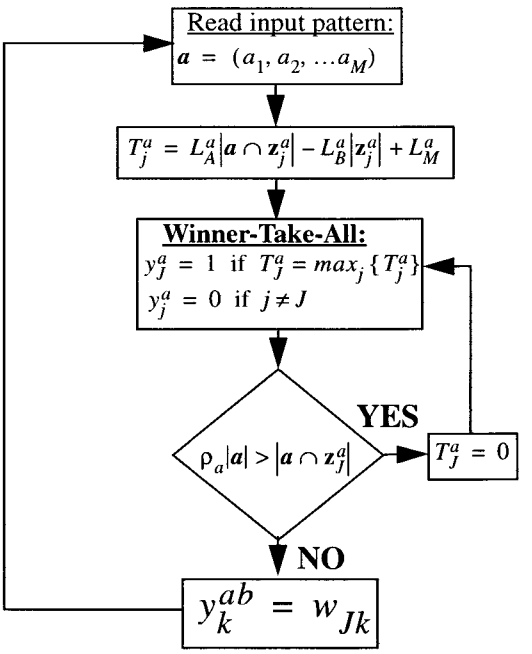


Fig. 5.13: Flow diagram of the prediction ARTMAP operation

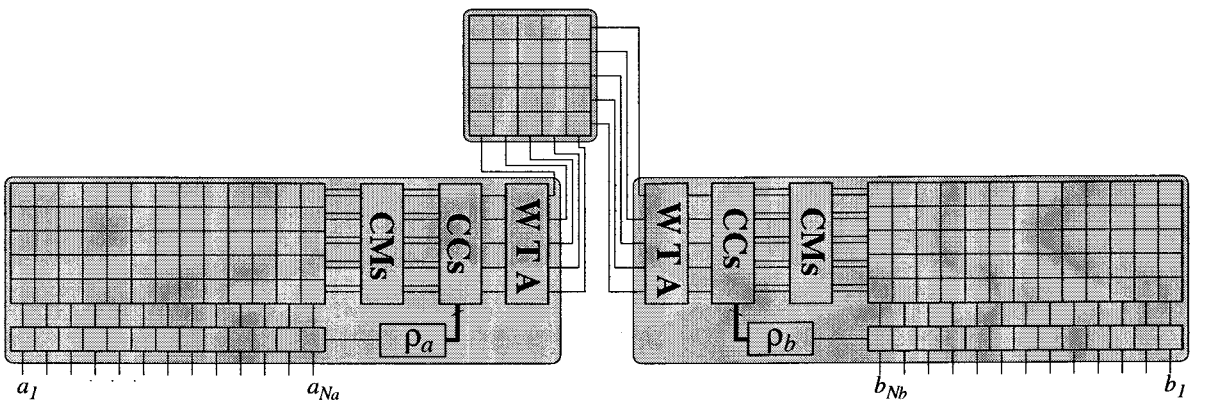


Fig. 5.14: Interconnection Diagram of the ARTMAP System

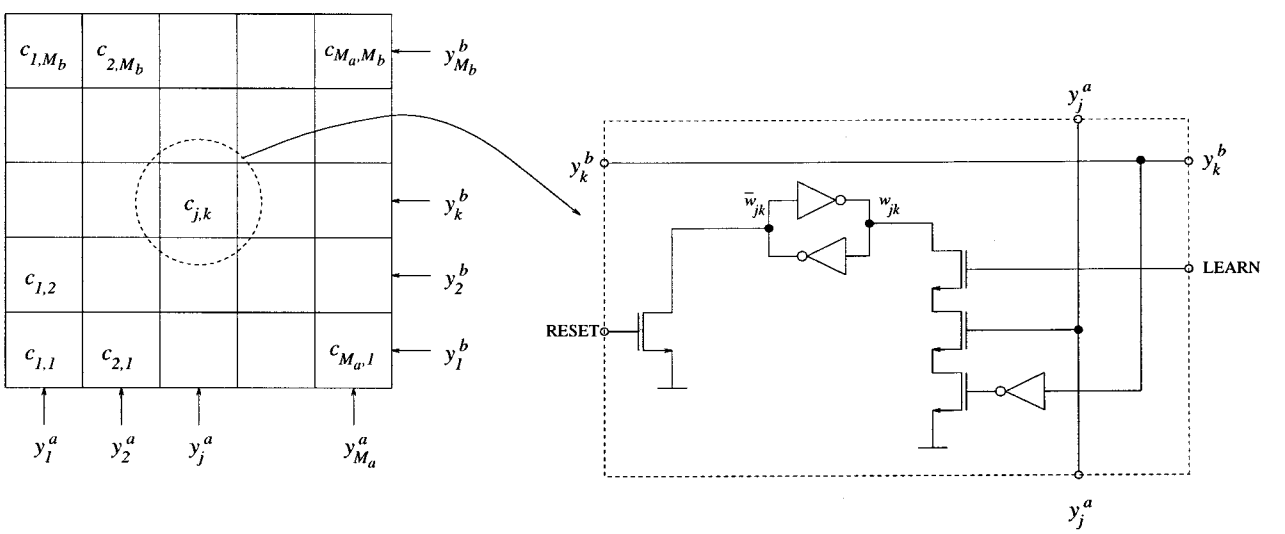


Fig. 5.15: Schematic of the Inter-ART Module

- The weights are initialized. Before applying the training sequence, a pulse is applied to the reset signal that makes all the weights w_{jk}, z_{ij}^{a} and z_{lk}^{b} be set to their high values.
- During the training mode:

  1.- Two input patterns **a** and **b** are presented to the ART 1 modules, which make a selection of categories u_J^a and u_K^b. These categories must satisfy the vigilance criterion imposed by vigilance parameters ρ_a and ρ_b set in the ART 1 modules.

  2.- After the activation of categories u_J^a and u_K^b, the value of w_{JK} is read out. Two alternatives may occur:

  2.1.- If $w_{JK} = 0$, a mismatch has occurred between the prediction of category u_J^a and category u_K^b. The *ART* 1^a vigilance parameter ρ_a is increased to ρ_a'=ρ_a+$\Delta\rho_a$ (where $\Delta\rho_a$ is the minimum ρ variation step in the ART 1 systems) and a new search process takes place in *ART* 1^a. This process is repeated until an *ART* 1^a category is found that makes $w_{JK} = 1$.

  2.2.- If $w_{JK} = 1$, a pulse is applied to the "LEARN" signal and the weights z_J^a, z_K^b and $\boldsymbol{w}_J$ are updated. The weights $\boldsymbol{w}_J$ corresponding to the active u_J^a category are updated to their new values given by,

$$w_{Jk}^{new} = \begin{cases} w_{Jk}^{old} & \text{if } y_k^b = 1 \\ 0 & \text{if } y_k^b = 0 \end{cases}, \tag{5.24}$$

  that is,

$$w_{Jk}^{new} = w_{Jk}^{old} y_k^b \tag{5.25}$$

  or in vector notation,

$$\boldsymbol{w}_J^{new} = \boldsymbol{w}_J^{old} \cap \mathbf{y}^b. \tag{5.26}$$

  Since only one y_k^b is equal to '1',

$$y_k^b = \begin{cases} 1 & \text{if } k = K \\ 0 & \text{otherwise} \end{cases}, \tag{5.27}$$

  it holds that,

$$w_{Jk}^{new} = \begin{cases} 1 & \text{if } k = K \\ 0 & \text{otherwise} \end{cases}. \tag{5.28}$$

  3.- The *ART* 1^a vigilance parameter ρ_a' is restored to its original value ρ_a when the input patterns **a** and **b** are substituted by a new pair of input patterns **a'** and **b'**. Then, a new selection process begins in *ART* 1^a and *ART* 1^b.

- During the prediction mode:

1.- An input pattern **a** is applied to the $ART\ 1^a$ module and a category u_j^a which satisfies the $ART\ 1^a$ vigilance criterion is chosen.

2.- The output of the inter-ART module is given by,

$$y_k^{ab} = \sum_j w_{jk} y_j^a = w_{JK} \tag{5.29}$$

Since for each J there is only one K such that $w_{JK} = 1$, let us call it w_{JK}. Therefore, category u_K^b is considered to be the prediction made by the ARTMAP system.

C. Experimental Results

The system level operation of the ARTMAP system has been also tested using the HP82000 digital test equipment. Fig. 5.16 and Fig. 5.17 show a test sequence performed on the system.

Fig. 5.16 shows a system training sequence. The first column, named "pattern_a", represents the input patterns applied to the $ART\ 1^a$ chip. Each input pattern is a different representation of one of the five vowels. The column named "pattern_b" represents the input patterns applied to the $ART\ 1^b$ system. The $ART\ 1^b$ input patterns are the first five digits. The columns named "weights_a" and "weights_b" represent the stored weights in the $ART\ 1^a$ and $ART\ 1^b$ categories after the classification and learning of each input pattern pair. The underlined categories are the ones that remain active after the search process has finished, and these are the only ones that are updated with learning. Below each $ART\ 1^a$ category we indicate the minimum value of the vigilance parameter ρ_a' needed in the search process to chose this category as the winning one, thus avoiding a prediction error (the vigilance parameter ρ_a was increased in steps of $\Delta\rho_a = 1/32$). The last column shows the stored weight in the inter-ART module which represent the learned correspondence between the $ART\ 1^a$ and $ART\ 1^b$ categories. During the training sequence, the system learns to identify each set of different representations of a vowel with a different digit. The vigilance parameter ρ_a is set to '0' and the current ratio parameter $\alpha = 2$ ($L_A^a = 10\mu A$ and $L_B^a = 5\mu A$). For the $ART\ 1^b$ system, we set the same current levels, $L_A^b = 10\mu A$ and $L_B^b = 5\mu A$, and a vigilance parameter $\rho_b = 0.75$. For this vigilance parameter, the $ART\ 1^b$ chip forms a different category for each input pattern (it perfectly learns the set of input patterns), so we can identify each category with the corresponding input pattern.

During the prediction sequence depicted in Fig. 5.17, only an input pattern **a** is applied to the system. The first column shows the input patterns applied to the system. These input patterns are the result of applying a random noise to the training patterns. The underlined $ART\ 1^a$ category is the one chosen by this chip after the search process and the underlined $ART\ 1^b$ category is the one the $ART\ 1^a$ category has learned to predict. During this sequence the current levels are $L_A^a = 10\mu A$, $L_B^a = 5\mu A$. The vigilance parameter is $\rho_a = 0$. Only one prediction error occurs despite the amount of noise present in the input patterns.

5.4. References

[5.1] N. R. Strader and J. C. Harden, "Architectural Yield Optimization," in *Wafer Scale Integration*, E. E. Swartzlander, Jr. (Ed.), pp. 57-118, Kluwer Academic Publishers, Boston, 1989.

[5.2] P. E. Allen and D. R. Holberg, *CMOS Analog Circuit Design*. New York: Holt, Rinehart and Winston, Inc., 1987.

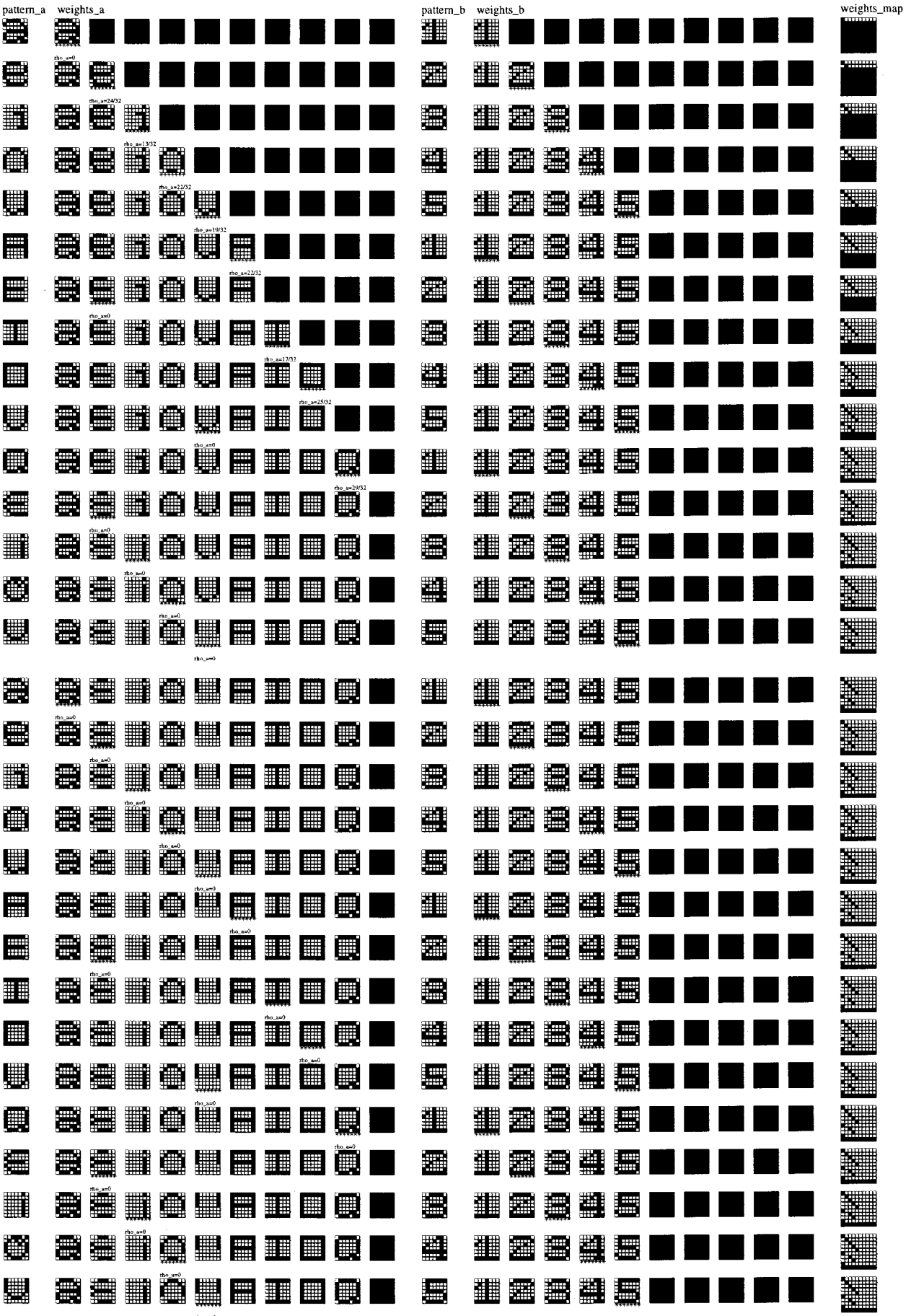


Fig. 5.16: Complete training sequence of the ARTMAP system for a vigilance parameters $\rho_a = 0$ and $\rho_b = 0.75$

[5.3] G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network," *Neural Networks*, vol. 4, pp. 565-588, 1991.

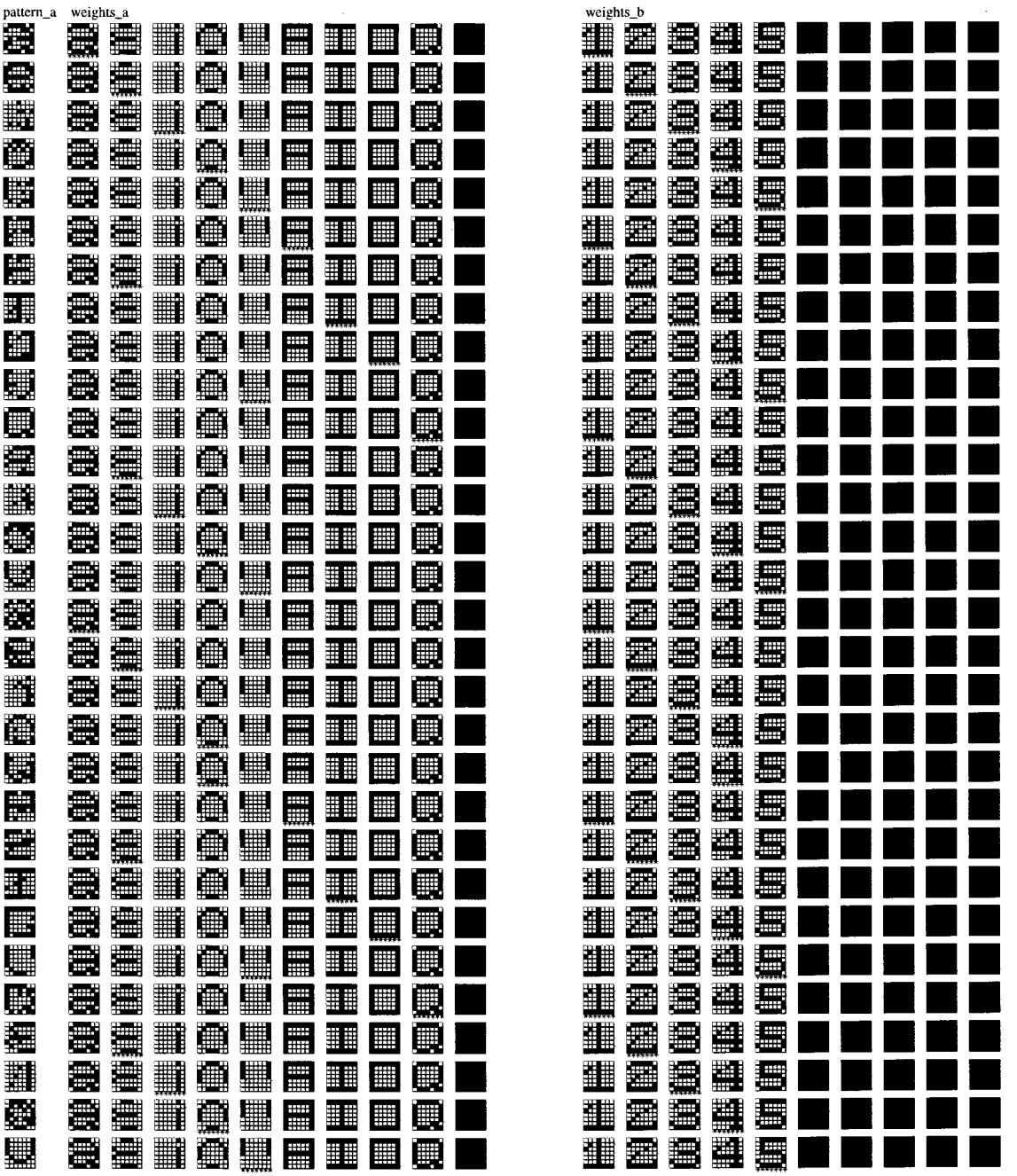


Fig. 5.17: Recognition sequence performed on a previously training ARTMAP system. The applied a input patterns are a noisy version of the a vectors of the training set. The vigilance parameter ρ_a is set to '0'

a TERESA SERRANO GOTARREDONA
CATEGORIZADORES NEURONALES EN VLSI

APTO CUM LAUDE

20 Diciembre 96