Síntesis automática de bloques de ganancia unitaria utilizando algoritmos genéticos

Por

Miguel Aurelio Duarte Villaseñor
Lic. en Electrónica

Tesis sometida como requisito parcial para obtener el grado de

Maestro en Ciencias en la especialidad de Electrónica

en el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Agosto, 2007
Tonantzintla, Puebla

Supervisada por:

Dr. Esteban Tlelo Cuautle
Investigador Titular del INAOE

Síntesis automática de bloques de ganancia unitaria utilizando algoritmos genéticos

Miguel Aurelio Duarte Villaseñor

2007

Sumario

Se presenta un nuevo método de representación genética para sintetizar bloques de ganancia unitaria tales como: seguidores de voltaje (VF), seguidores de corriente (CF), espejos de voltaje (VM) y espejos de corriente (CM). El método de síntesis se realizó utilizando algoritmos genéticos (GAs), los cuales son técnicas de búsqueda basadas en los mecanismos de selección natural y en la genética biológica.
Se muestra la base para realizar automáticamente el netlist de circuitos integrados mediante un código llamado 'cromosoma' que es dividido en cuatro genes: gen de pequeña señal (genSS), gen de síntesis del MOSFET (genSMos), gen de polarización (genBias) y gen de síntesis de espejos de corriente (genCM). Este último para sintetizar las fuentes de corriente ideales, utilizadas en la polarización de los circuitos por espejos de corriente CMOS. Los genes crecen según el número de elementos nullor usados para modelar el comportamiento de los circuitos.
El método de síntesis propuesto se ha programado en MatLab; y este utiliza T-SPICE para evaluar el comportamiento de las topologías en el nivel de abstracción de transistor. De esta manera, el método selecciona los circuitos más apropiados por elitismo.
Finalmente, se muestra la aplicación del método descrito para sintetizar topologías de circuitos de ganancia unitaria. Asimismo, se describe la evolución de los bloques de ganancia unitaria para sintetizar circuitos más complejos como los current conveyors.

Palabras Clave: Diseño Electrónico Automatizado, algoritmo genético, nullor, síntesis de circuitos, seguidor de voltaje, seguidor de corriente, espejo de voltaje, espejo de corriente, current conveyor.

Automatic synthesis of unity-gain blocks using genetic algorithms

Miguel Aurelio Duarte Villaseñor

2007

Abstract

A new genetic representation method is introduced to synthesize unity-gain blocks such as: voltage followers (VF), current followers (CF), voltage mirrors (VM), and current mirrors (CM). The synthesis method was implemented using genetic algorithms (GAs), which are search techniques based in the mechanisms of natural selection and the biological genetics.
It is shown the guidelines to automatically realize the netlist of integrated circuits through a code called 'chromosome', that is separated in four genes: gene of small signal (genSS), gene of synthesis of the MOSFET (genSMos), gene of bias (genBias), and gene of synthesis of current mirrors (genCM). The last one is used to synthesize the ideal current sources used in the biasing of the circuits by CMOS current mirrors. The genes grow according to the number of elements nullor used to model the behavior of the circuits.
The proposed synthesis method has been programmed in MatLab, and it uses T-SPICE to evaluate the behavior of the topologies in the transistor level of abstraction. In this way, the method selects the most appropriate circuits by elitism.
Finally, it is shown the application of the method to synthesize unity-gain circuit topologies. Also, the evolution of the unity-gain blocks is described to synthesize more complex circuits such as current conveyors.

Keywords: Electronic Design Automation, genetic algorithm, nullor, circuit synthesis, voltage follower, current follower, voltage mirror, current mirror, current conveyor.

Dedicatorias

A mi esposa, Johana M. García Ortega,
Por tantas demostraciones de amor
en todos los años que tenemos de conocernos.
Hasta que la muerte nos separe.

Agradecimientos

Mi agradecimiento al Dr. Esteban Tlelo Cuautle por sus consejos, su paciencia y por todo el apoyo brindado para el desarrollo de esta tesis, gracias.

Muchas gracias a mi familia, a mi esposa, a mis amigos y a todas las personas que me apoyaron moralmente en la culminación de esta tesis.

Este trabajo fue financiado por CONACyT con la beca para estudios de maestría #199283 y la tesis forma parte del proyecto número 48396-Y. “Electrónica evolutiva: Síntesis automática de circuitos integrados analógicos”, aprobado en 2006.

Contenido

Capítulo 1

Introducción.

1.1 Antecedentes.

Hoy en día, las herramientas clave para manipular la complejidad en el proceso de diseño de Circuitos Integrados CMOS son generadas por la industria del Diseño Electrónico Automático (EDA). Este tipo de herramientas permiten eliminar ambigüedades dentro del proceso de diseño ya que pretenden representarlo como una metodología estructurada [1]. El EDA se usa para automatizar muchas de las tareas que son de rutina y repetitivas en el diseño analítico [1].

El objetivo del EDA se orienta a reproducir y optimizar el diseño manual, aprovechando las tareas repetitivas que se realizan en el proceso de diseño. En algunos casos en el diseño de sistemas analógicos se pueden identificar bloques funcionales tales como amplificadores, seguidores de voltaje, espejos de corriente, entre otros; los cuales pueden repetirse y además pueden sintetizarse automáticamente aplicando metodologías del EDA [1-5]. De esta manera, las herramientas EDA incrementan la productividad en el diseño, aun para bloques de circuitos que no sean repetitivos. Por ello, el EDA analógico y la automatización de circuitos tienen un papel importante en el proceso de diseño [1,3]. Sin embargo, la automatización del diseño analógico es más complejo comparado con el diseño digital, porque es más heurístico y menos sistemático. Además no ha sido posible todavía para los diseñadores automatizar todos los niveles de abstracción y la síntesis de circuitos desde un diseño de alto nivel. Esto se debe principalmente a que el diseño analógico requiere de

experiencia, intuición y creatividad, ya que se trabaja con un gran número de parámetros y de algunas interacciones complejas entre ellos. Asimismo, debido a la gran variedad de topologías de circuitos analógicos, se busca de alguna manera automatizar ciertas tareas de diseño, tantas como sea posible [1-3].

Es una realidad que muchos de los CIs están compuestos por bloques que son repetitivos como inversores, espejos de corriente, pares diferenciales, etc.; por lo tanto, el diseño de estos circuitos se puede automatizar para explorar la búsqueda de nuevos circuitos y además para reducir el tiempo de diseño [1].

El diseño de circuitos analógicos es muy adaptable para técnicas evolutivas. Como se muestra en [3,5-12], la síntesis analógica es un problema que se esta trabajando. Los algoritmos genéticos (GAs) son usados para la síntesis analógica, ellos tienen la descripción de alto nivel y operan sobre el principió de ‘sobrevivencia del más apto’. Así que ellos tienen la capacidad de generar nuevos diseños de solución para una población de soluciones existentes y descartan las soluciones que tienen una inferior medida de aptitud.

Con el fin de automatizar el proceso de síntesis de circuitos analógicos y de obtener nuevas topologías de circuitos de ganancia unitaria [13]; en este trabajo se desarrolla una herramienta basada en la aplicación de algoritmos genéticos (GAs) [3-5,14]. El método propuesto es capaz de sintetizar bloques de ganancia unitaria: seguidores de voltaje (VFs) [15], seguidores de corriente (CFs) [2], espejos de voltaje (VMs) [16] y espejos de corriente (CMs) [17]; cuyas ecuaciones características y símbolos se muestran en la figura 1.1. Cabe mencionar que la mayor contribución en este trabajo es la propuesta de un nuevo método para codificar y decodificar circuitos modelados con anuladores [17] usando una cadena de bits llamada cromosoma.

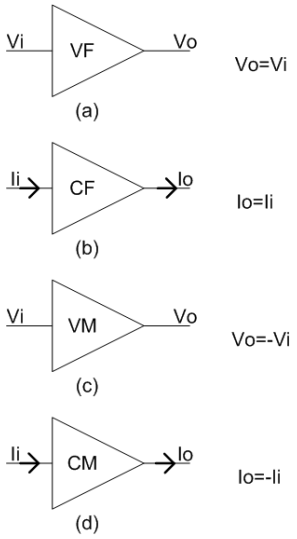

Figura 1.1. (a) Seguidor de voltaje, (b) Seguidor de corriente, (c) Espejo de voltaje, (d) Espejo de corriente.

1.2 Objetivos.

El objetivo principal de este trabajo es implementar un método de síntesis automática de bloques de ganancia unitaria. Para esto se desarrolla un sistema inteligente basado en algoritmos genéticos para sintetizar topologías de VFs, CFs, VMs y CMs. Como objetivos secundarios se tiene:

- Proponer un nuevo método de codificación usando anuladores.
- Encontrar nuevas topologías de VFs, CFs, VMs y/o CMs.
- Evolucionar los bloques de ganancia unitaria, ya sea por conexión o súper-imposición para formar circuitos current conveyors (CC).
- Obtener nuevas topologías de CCII+, CCII-, ICCII+ e ICCII-.

1.3 Motivación.

Actualmente existen muchas aplicaciones de opamps y current conveyors (CC) [18-28]; esas aplicaciones son factibles de implementarse usando bloques de ganancia unitaria VF, VM, CM y CF; debido a que la mayoría de los CCs se diseñan con estos bloques [29,30]. En la actualidad, no hay métodos automáticos para el diseño de bloques de ganancia unitaria. De esta manera, el diseño de un sistema que permita la síntesis automática

de bloques de ganancia unitaria y CCs facilitará la creación de nuevas topologías de circuitos analógicos. Asimismo, en los últimos años se han presentado métodos para la síntesis de circuitos analógicos aplicando sistemas inteligentes, en su mayoría son enfocados al diseño de circuitos pasivos o circuitos con amplificadores operacionales [5-12].

En resumen, se realiza este trabajo con el fin de desarrollar una herramienta que sea capaz de encontrar nuevas topologías de circuitos VFs, CFs, VMs y CMs y que estos bloques de ganancia unitaria puedan evolucionar a circuitos más complejos como los CCs.

1.4 Organización de la tesis.

En esta sección se proporciona un panorama general de la organización de este trabajo; presentando una vista general de lo que se aborda en cada uno de los capítulos.

En el capítulo 2 se describe brevemente los conceptos de algoritmos evolutivos (EA), destacando los algoritmos genéticos (GAs) que son los utilizados para realizar este trabajo.

En el capítulo 3 se explica como se propone describir los circuitos por medio de genes y como estos genes forman cromosomas; de esta forma se puede realizar un algoritmo genético que trabaje con cromosomas, sintetizándolos posteriormente en circuitos con MOSFETs. En este capítulo se describe como es el diagrama de flujo del algoritmo genético realizado.

La propuesta descrita en el capítulo 3 se programó en MatLab. En el capítulo 4 se exhibe la utilización del método de síntesis y se muestran algunos de los circuitos obtenidos y el comportamiento del algoritmo genético. Por último, se describe como pueden evolucionar los cromosomas para convertirse en circuitos más complejos como CCs.

Las conclusiones que se obtuvieron en la realización de esta tesis se muestran en el capítulo 5.

En el apéndice A se muestran las publicaciones realizadas por el autor.

En el apéndice B se muestran descritos los archivos utilizados por SPICE en las simulaciones, explicando un netlist realizado por el sistema; y se muestran los parámetros de la tecnología de 0.35µm utilizada.

Capítulo 2

Algoritmos genéticos.

2.1 Introducción.

Los organismos vivos poseen una destreza consumada en la resolución de problemas de adaptación al medio que los rodea; y manifiestan una versatilidad de soluciones capaz de ridiculizar a los programas para computadora más sofisticados. Esta observación resulta un tanto molesta para quienes se dedican a la informática o a resolver problemas de reproducir el comportamiento de seres vivos, que han de dedicar meses o años de esfuerzo intelectual a preparar un algoritmo o una solución, mientras los organismos obtienen sus habilidades (de adaptación, funcionamiento, comportamiento) a través de mecanismos como la evolución y la selección natural [31,32].

De acuerdo al modelo de Darwin de la evolución de las especies, toda la vida en nuestro planeta puede ser explicada a través de un conjunto de procesos estadísticos, que actúan en y dentro de las poblaciones, individuos y especies; como por ejemplo: la reproducción, herencia, mutación, competencia y selección de los organismos [31].

La selección natural es el proceso donde los individuos mejor adaptados al medio ambiente tienen mayor probabilidad de producir descendientes; que aquellos que son menos aptos. Adicionalmente a las teorías de Darwin, son necesarios los planteamientos de Mendel y de la Genética Matemática para comprender cómo estos procesos ayudan en la selección natural [31,32].

El principio de supervivencia del más apto, es el eje central sobre el cual se desarrollan técnicas de aprendizaje que responden al nombre genérico de Algoritmos Evolutivos (EAs). Los EAs simulan el proceso evolutivo en una computadora, con la finalidad de resolver problemas de aprendizaje, búsqueda, clasificación u optimización. El problema a resolver puede caer dentro de una gran variedad de disciplinas, incluyendo la Biología, Ingeniería, Servicios Financieros y Ciencias Computacionales. Las tres técnicas evolutivas principales son: las estrategias evolutivas (ESs), la programación genética (GPs) y los algoritmos genéticos (GAs) [5].

Uno de los primeros argumentos lógicos que validan el uso de la teoría de la evolución para desarrollar una metodología de solución de problemas, es la observación del proceso de evolución que ha optimizado el proceso biológico. Por lo tanto, la aplicación del modelo evolutivo y de selección natural a un conjunto de posibles soluciones para un problema, debería producir una buena (u óptima, en el mejor de los casos) solución para este problema [31-34].

2.2 Algoritmos Evolutivos.

El término de Computación Evolutiva es relativamente joven, el cual fue inventado en 1991 y representa un gran esfuerzo para acercar a los investigadores a seguir los diferentes aspectos de la evolución [33]. Dentro de las principales técnicas de la computación evolutiva se encuentran: los Algoritmos Genéticos, las Estrategias Evolutivas y la Programación Genética, las cuales tienen en común que cada una de ellas se apoyan en la reproducción, la variación aleatoria, la competición y la selección de individuos contendiendo dentro de una población [3-5,14], lo cual es en sí la esencia de la evolución, cabe aclarar que la evolución es un proceso de optimización, el cual no implica la perfección.

Este proceso de optimización puede buscar soluciones altamente precisas a ciertos problemas en particular. Por lo tanto podemos describir a la evolución en términos de un algoritmo evolutivo que puede ser usado para solucionar algunos problemas de optimización en ingeniería [33]. En

este trabajo se aplican los algoritmos genéticos para generar bloques de ganancia unitaria a partir de una descripción basada en un código genético binario, que se describirá en el siguiente capítulo.

2.3 Algoritmos Genéticos.

Como se menciona en párrafos anteriores, dentro de las principales técnicas de Computación Evolutiva encontramos los GAs, las ESs y la GP, las cuales en la actualidad son consideradas como unas poderosas herramientas de optimización [5].

Los GAs son técnicas estocásticas de búsqueda basadas en mecanismos de selección natural y genética [3-5,14,31-34]. Los primeros conceptos de los GAs fueron desarrollados en la Universidad de Michigan por Holland en 1967, posteriormente publicando su libro “Adaptation in Natural and Artificial Systems” [34]. Los GAs son métodos adaptivos que pueden ser utilizados en las búsquedas y obtención de soluciones a problemas de optimización. Estos algoritmos están basados en los procesos genéticos de los organismos biológicos, codificando una posible solución a un problema a través de un código nombrado *cromosoma*.

Los cromosomas representan individuos que son llevados a lo largo de varias generaciones, evolucionando de acuerdo a los principios de selección natural y supervivencia del más apto, lo cual fue descrito por primera vez por Charles Darwin [31-34]. Los GAs utilizan una analogía directa del fenómeno de evolución en la naturaleza.

Comienzan con un conjunto inicial de soluciones aleatorias llamado *población inicial*. Cada individuo en la población es llamado cromosoma y representa una posible solución al problema.

Un cromosoma es una cadena de símbolos llamados genes; que por lo general son representados por una cadena de dígitos binarios. Los cromosomas evolucionan a través de iteraciones llamadas generaciones. En cada generación los cromosomas son evaluados, usando una medida de aptitud. A los más aptos se les da la oportunidad de reproducirse mediante recombinaciones con otros individuos de la población,

produciendo descendientes con características de ambos padres. Los miembros menos adaptados poseen pocas probabilidades de que sean seleccionados para la reproducción, por lo tanto sus características desaparecen.

La siguiente población, llamada descendencia, se forma combinando dos cromosomas de la generación actual usando el operador de cruza o modificando un cromosoma usando un operador de mutación o realizando una combinación de ambas operaciones [3-5,14,32-34]. Esta nueva generación contiene una proporción más alta de las características poseídas por los mejores miembros de la generación anterior. De esta forma, a lo largo de varias generaciones las mejores características son difundidas a lo largo de la población mezclándose con otras. Favoreciendo la recombinación de los individuos mejor adaptados, con esto es posible recorrer las áreas más prometedoras del espacio de búsqueda. Si el GA ha sido diseñado correctamente, la población convergerá a una solución óptima o casi óptima al problema [5,33].

Generalmente, los GAs son implementados siguiendo el siguiente ciclo:

- Se genera la población inicial de manera aleatoria.
- Se evalúa la aptitud de todos los individuos de la población.
- Se crea una nueva población mediante los operadores de cruza y mutación.
- Se itera, utilizando la nueva población hasta que se encuentra una solución satisfactoria.

En la figura 2.1 se muestra la estructura general de los GAs, donde se pueden apreciar los cuatro pasos descritos anteriormente.

El poder del GA proviene del hecho de que la técnica es robusta y puede ser utilizada exitosamente en un amplio rango de problemas como: la optimización de funciones numéricas, transportación, localización, en la optimización de la velocidad de herramientas e incluso en algunos problemas que son difíciles de resolver por otros métodos [5,14,32-34]. Los GAs no garantizan que encontrarán la solución óptima al problema

pero son generalmente buenos encontrando soluciones aceptables a problemas en corto tiempo.

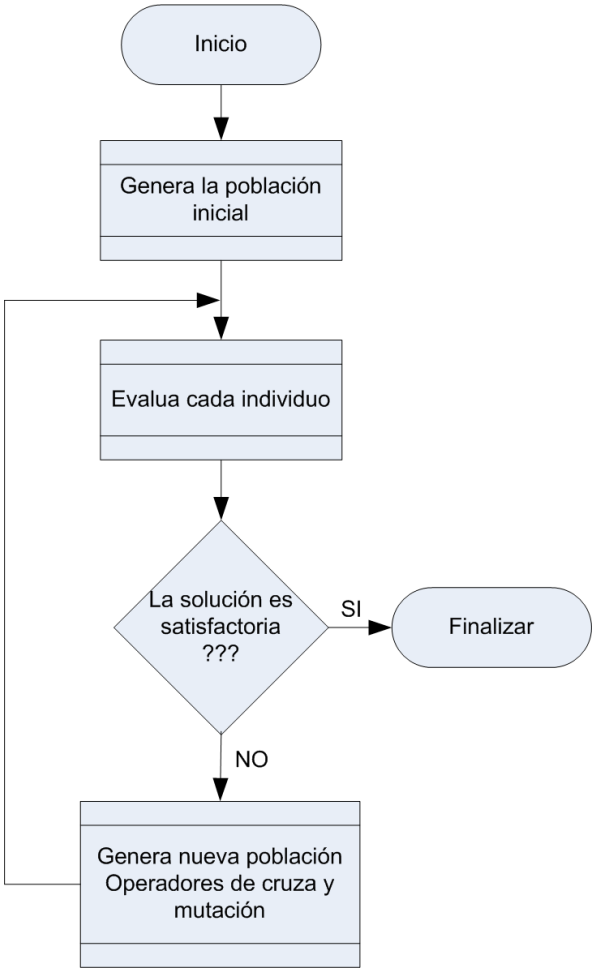

Figura 2.1. Estructura general de los algoritmos genéticos.

Algunas características de los GAs son [3-5,14,32-34]:

- Trabajan con un conjunto de soluciones y no con las soluciones mismas.
- Buscan la solución en una población de posibles soluciones y no con soluciones solitarias.
- Utilizan una función de evaluación, no utilizan funciones derivadas de esta o algún otro método o conocimiento auxiliar.
- Utilizan reglas de transición probabilística y no determinística.

- Combina elementos de búsqueda estocástica y directa que hace un balance notable entre exploración y aprovechamiento del espacio de búsqueda.

Dado un individuo, la función de evaluación consiste en asignarle un valor de aptitud, el cual es proporcional a la utilidad o habilidad del individuo representado. En muchos casos el desarrollo de una función de evaluación involucra hacer una simulación; en otros casos la función puede estar basada en el rendimiento y representar sólo una evaluación parcial del problema. Adicionalmente debe ser rápida, ya que hay que aplicarla para cada individuo de cada población en las generaciones sucesivas. Por lo cual, gran parte del tiempo de corrida del algoritmo genético se emplea en la función de evaluación. Además existe un problema en los algoritmos genéticos debido a una mala formulación del modelo, en el cual los genes de unos pocos individuos relativamente bien adaptados, pero no óptimos, puede rápidamente dominar la población, causando que converja a un mínimo local. Una vez que esto ocurre, la habilidad del modelo para buscar mejores soluciones es eliminada completamente, quedando sólo la mutación como vía para buscar soluciones alternativas; y el algoritmo se convierte en una búsqueda lenta al azar [3-5,14,32-34].

2.4 Genotipo y Fenotipo.

Si un problema puede ser representado por un conjunto de parámetros (genes), y estos pueden ser unidos para formar una cadena de valores; a esta cadena se le llama cromosoma y a este proceso se le conoce como codificación [33].

Es común que la representación de individuos a través de cromosomas se haga con cadenas de dígitos binarios, tal representación es llamada **genotipo**; entonces, es necesaria la conversión del genotipo a los valores que pertenecen a un individuo, la cual es llamada **fenotipo** [34].

Existen varios aspectos relacionados con la codificación de un problema a ser tomados en cuenta en el momento de su realización:

- Se debe de usar la representación más pequeña de los parámetros, normalmente se utiliza una representación binaria.
- Las variables que representan los parámetros del problema deben ser discretas para poder representar cadenas de bits.
- La mayor parte de los problemas tratados con algoritmos genéticos son no lineales y muchas veces existen relaciones entre las variables que conforman las soluciones.
- El tratamiento de los genotipos inválidos debe ser tomado en cuenta para el diseño de la codificación.

En este trabajo de tesis se sintetizan bloques de ganancia unitaria donde los genotipos son compuestos por cadenas de bits, llamados cromosomas. Estos cromosomas sintetizan un circuito según una codificación propuesta en el capítulo 3. Estos circuitos realizados con transistores son los llamados fenotipos en los GAs.

2.5 Operaciones de cruza y mutación.

Los operadores genéticos son los diferentes métodos u operaciones que se pueden realizar sobre una población en los GAs, estos se dividen en cuatro categorías:

- Selección
- Cruza o Recombinación
- Mutación
- Reemplazo o Reinserción

En este trabajo no se profundizan en forma detallada cada uno de estos operadores, ya que algunas de sus características se pueden revisar en las referencias [3-5,14,32-34] entre otras.

Los dos procesos que contribuyen a la evolución son la cruza y la mutación. Esta última juega un papel importante, pero el determinar que tan significativa sea su aplicación es un tema de debate, ya que al ser utilizada demasiado el GA se puede convertir en una búsqueda al azar [5].

2.5.1 Operador de cruza.

La cruza o recombinación consiste en unir de alguna forma los cromosomas de dos padres para formar 1 o más descendientes. Existen diversas variaciones dependiendo del número de puntos de división a emplear, de la forma de ver el cromosoma, etc. [32-34]. Dependiendo de la representación de las variables en los cromosomas se pueden aplicar los siguientes métodos:

Recombinación de parámetros reales:

- Recombinación Discreta.
- Recombinación Intermedia.
- Recombinación Lineal.
- Recombinación Lineal Extendida.

Recombinación de parámetros binarios:

- Cruza de Punto Sencillo.
- Cruza de Múltiples Puntos.
- Cruza Uniforme.
- Cruza Aleatoria.
- Cruza con Reducción de Sustituto.

Estos métodos se describen en la referencia [33]. La operación de recombinación utilizada en este trabajo es la llamada cruza de múltiples puntos, la cual se puede comprender de una manera sencilla observando la figura 2.2.

La figura 2.2 muestra dos cromosomas padres, estos están formados por cuatro genes; cada uno de estos genes representa una característica de un circuito (cada gen esta explicado en el capítulo 3). Al realizar la operación de cruza se tienen 14 posibles combinaciones, estos cromosomas hijos son las uniones de los cuatro genes de los padres. Es decir, estos dos padres pueden generar hasta 14 cromosomas hijos. Para conservar un cierto tamaño de población de estos posibles 14 hijos solo son generados de 2 a 4 cromosomas; realizando la obtención de estos al azar.

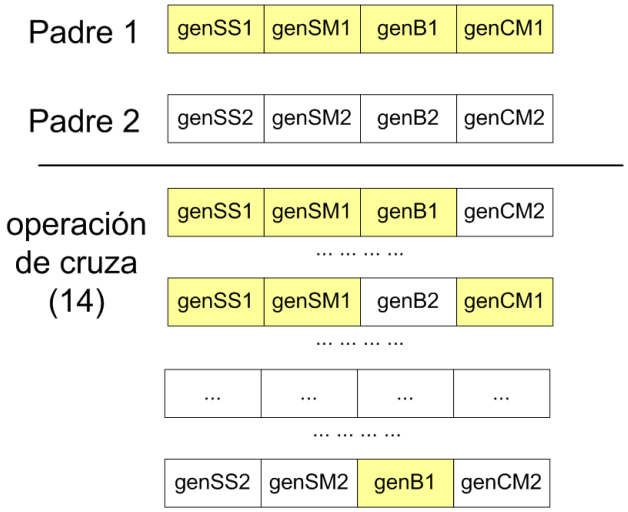

Figura 2.2. Operación de cruza de múltiples puntos.

2.5.2 Operador de mutación.

La mutación se encarga de modificar en forma aleatoria uno o más genes del cromosoma de un descendiente. La representación de las variables determinara el método de mutación [32-34]; así, dos de los principales métodos son los siguientes:

- Mutación de Parámetros Binarios.
- Mutación de Parámetros Reales.

Después de la cruza de los descendientes se experimenta mutación. Las variables de los descendientes son mutadas por la adición de pequeños valores aleatorios con la probabilidad baja, esto llamado ‘el tamaño del paso de mutación’ en algunas referencias [3-5,14]. La probabilidad de mutar una variable es puesta para ser inversamente proporcional al número de variables o dimensión del cromosoma [32-34].

La operación de mutación de parámetros binarios consiste en cambiar uno o más bits de un cromosoma y es el tipo de mutación utilizada en este trabajo. En la figura 2.3 se muestra un cromosoma compuesto por cuatro genes; cada gen esta compuesto de dos o cuatro bits, al aplicar el operador de mutación uno de estos bits cambia de ‘0’ a ‘1’ ó de ‘1’ a ‘0’.

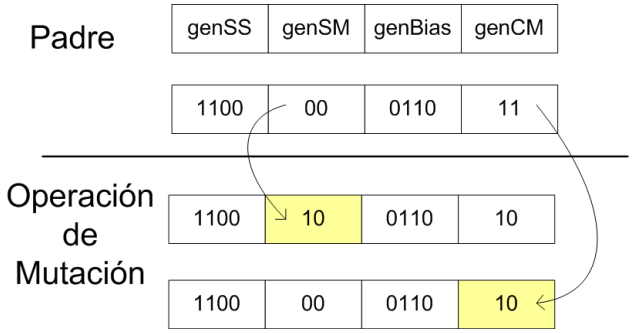

Figura 2.3. Operación de mutación de parámetros binarios.

Capítulo 3

Método de síntesis propuesto.

3.1 Introducción.

Existen trabajos previos acerca de herramientas de síntesis de circuitos y sistemas analógicos como IDAC, OPASYN, OASYS, O-ISAAC, STATIC, ARCHGEN, DARWIN, ANGIE o ANACONDA recopilados en [1]. Algunos de estos dimensionan los MOSFET de los circuitos, otros mejoran algún parámetro en los dominios de CD, CA y/o tiempo de un cierto circuito y otros realizan filtros o encuentran funciones analógicas con circuitos CMOS. También existen trabajos desarrollados con GAs; estos muestran una codificación de circuitos en un cromosoma como se muestra en los trabajos de Wim Kruiskamp y Domine Leenaerts reimpreso en [1], Torres [2], Mattiussi [3], entre otros [5]. Sin embargo, estas descripciones se enfocan a optimizar un circuito en CD, CA y/o respuesta en el tiempo por lo que sus codificaciones binarias representan dimensiones de sus transistores y no como están realizadas estas topologías. Así, que estas codificaciones no son útiles para trabajar en esta tesis y se tiene la necesidad de realizar una codificación diferente a las ya mencionadas.

En este capítulo se presenta la propuesta de un algoritmo genético para la síntesis de circuitos de ganancia unitaria [2,15]. De esta manera, la primera tarea a resolver es la decodificación de un circuito a través de un cromosoma (o genotipo). Para decodificar un circuito primero se presenta la descripción del elemento anulador [17] y sus propiedades. Posteriormente se describe el proceso de búsqueda de una solución a través de un GA.

3.2 El elemento anulador.

En [13] se encuentra una recopilación de metodologías de análisis, diseño y síntesis de circuitos analógicos usando anuladores (nullors). Hanspeter Schmid [35] demostró su utilidad para modelar el comportamiento ideal de varios dispositivos activos, tales como: amplificador operacional (opamp), amplificador operacional de transconductancia (OTA), current conveyor (CC), etc. El nullor es un dispositivo ideal de dos puertos con cuatro variables asociadas v_{nu}, i_{nu} en el puerto de entrada y v_{no}, i_{no} en el puerto de salida, como se muestra en la figura 3.1. Este dispositivo es un elemento compuesto por un elemento O (nullator) y por un elemento P (norator). El voltaje y la corriente de nullator son siempre cero; para el norator el voltaje y la corriente son arbitrarios [2,13,15,16]. Las propiedades ideales del nullor se describen en la ecuación (3.1).

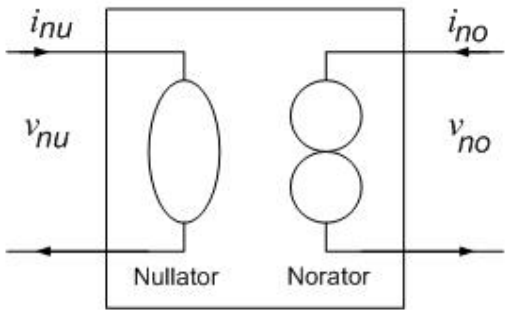

Figura 3.1. Nullor.

$$\begin{bmatrix} v_{nu} \\ i_{nu} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_{no} \\ i_{no} \end{bmatrix} \quad (3.1)$$

Las propiedades de los elementos O y P se pueden aprovechar para modelar el comportamiento ideal de los bloques de ganancia unitaria. Para modelar el comportamiento ideal del VF se puede utilizar uno, dos o un puente de cuatro elementos O como se muestra en la figura 3.2

[2,15,16]; donde utilizando (3.1) se demuestra que el voltaje a la salida del VF (Vo) es igual al voltaje de entrada (Vi). Dado que el voltaje y la corriente del nullator son igual a cero, el puente de cuatro nullators modela el comportamiento de uno simple.

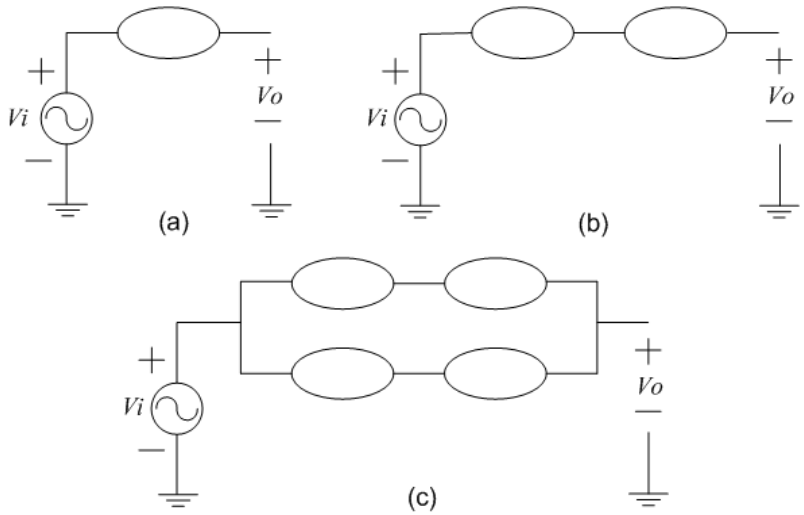

Figura 3.2. Modelos de VFs usando nullators.

Para formar un CF se utilizan celdas genéricas formadas por uno, dos y un puente de cuatro norators como se muestra en la figura 3.3 [2,17]. En estas celdas las variables eléctricas son i_i e i_o. Dado que, la corriente que circula a través del norator es la misma, $i_o = i_i$ para el CF.

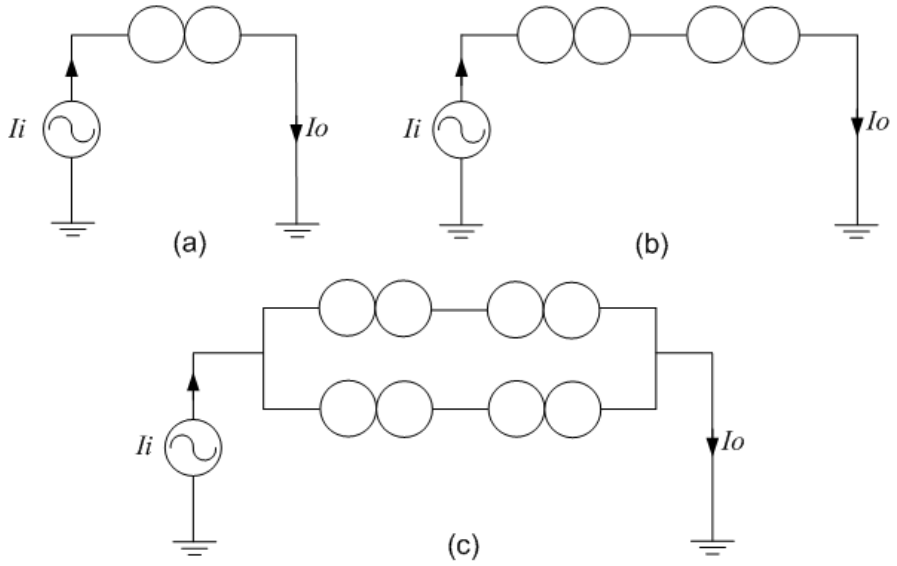

Figura 3.3. Modelos de CFs usando norators.

En la figura 3.4 se muestra el modelo de un transistor MOS utilizando el nullor. De esta manera, la tarea a realizar es agregar norators a los circuitos que estén representados por nullators y agregar nullators a las topologías que estén representadas por norators con el fin de formar pares O-P; y con esto obtener la síntesis de un transistor MOS. Se observa en la figura 3.4 que el punto de unión O-P es considerado como el *source* (S), la terminal libre del elemento O es el *gate* (G) y la terminal libre del elemento P es el *drain* (D) del MOSFET.

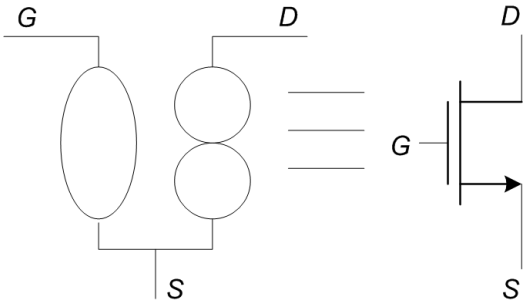

Figura 3.4. Modelo del MOSFET usando el nullor.

3.3 Codificación genética de un VF.

3.3.1 Genotipo de un VF.

Partiendo del modelo ideal de un VF utilizando solo un elemento O (figura 3.2.a), se procede a la generación de un circuito con transistores. Primero, para propósito de síntesis cada elemento O debe estar unido a un elemento P. De esta manera, un elemento P puede agregarse a un elemento O a la izquierda (nodo i), a la derecha (nodo j), o en paralelo (entre los nodos i y j); como se ilustra en la figura 3.5 [2,36]. Como puede inferirse, cada par O-P se puede representar por un gen de 2 bits de longitud; esta codificación crea el gen de pequeña señal (genSS), el cual se describe en la tabla 3.1.

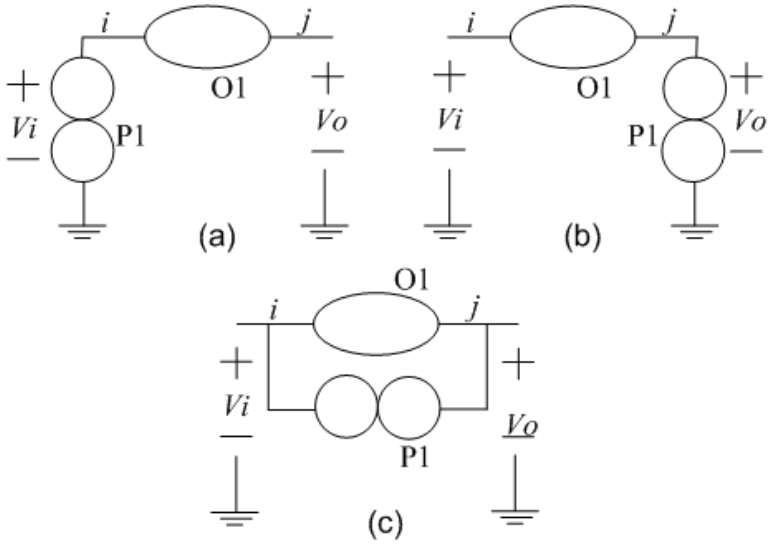

Figura 3.5. Adición de un elemento P: a) En el nodo i,
b) en el nodo j, y c) entre los nodos i y j.

Tabla 3.1. Codificación del genSS a partir de la figura 3.5.

a0	a1	Unión	
0	0	Pij	Punto de unión i (Figura 3.5.c)
0	1	Pi	(Figura 3.5.a)
1	0	Pj	(Figura 3.5.b)
1	1	Pji	Punto de unión j (Figura 3.5.c)

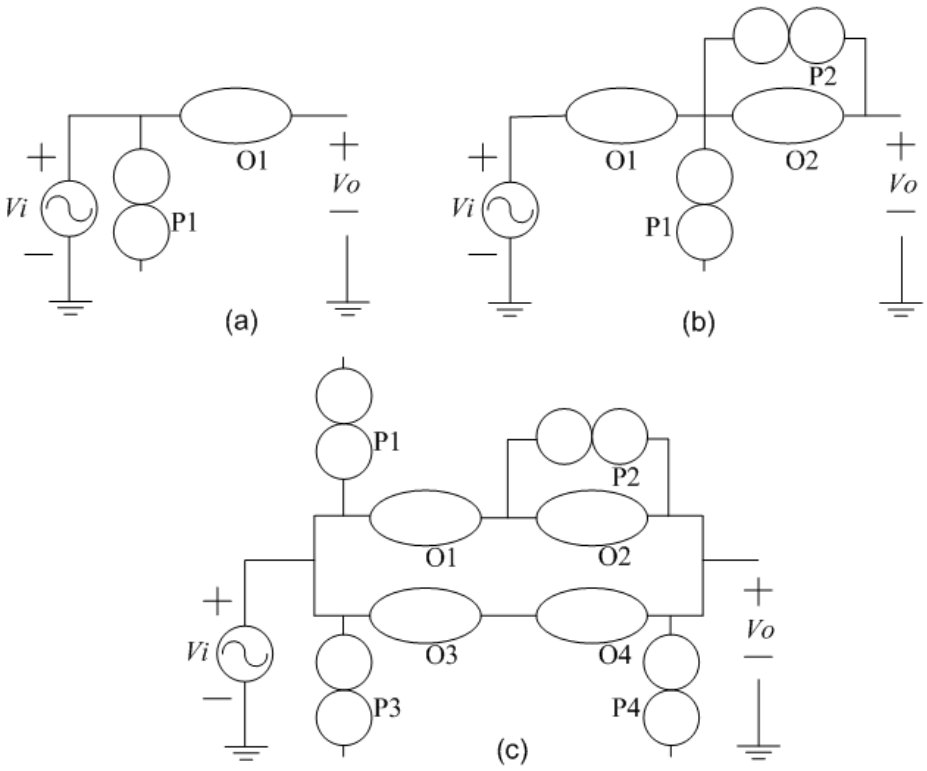

Figura 3.6. Ejemplos de VFs formados por pares O-P.

En la figura 3.6 se muestran ejemplos de generación de pares O-P a partir de la figura 3.2. Cada par O-P se sintetiza por un MOSFET como se mostró en la figura 3.4, en este proceso se utiliza un gen de síntesis del MOSFET (genSMos), el cual consiste de un bit (a2) para describir el tipo de transistor, así cuando genSMos es cero se sintetiza un N-MOS y cuando es uno se genera un P-MOS (tabla 3.2).

Tabla 3.2. Codificación del genSMos.

a2	Tipo de transistor
0	N-MOS
1	P-MOS

Para cada circuito sintetizado con MOSFETs, primero se agregan dos fuentes de voltaje nombradas Vdd y Vss para alimentar el circuito con niveles positivos y negativos, respectivamente. Posteriormente se procede a agregar fuentes de corriente utilizando un gen de polarización (genBias); este contiene dos bits (a3a4). La combinación 00 significa que el drain del MOSFET se conectara a Vdd mientras que el source se conectará a una fuente de corriente conectada a Vss. El 01 significa que el drain se conecta a Vss y el source a una fuente de corriente conectada a Vdd; las otras dos combinaciones (10 y 11) significan que tanto al drain como al source se les conectaran fuentes de corriente a Vdd y a Vss; esto se resume en la tabla 3.3.

Tabla 3.3. GenBias: Bits de asignación para agregar fuentes de corriente.

genBias	Conexión	
a3a4	*Drain*	*Source*
0 0	Vdd	Iss
0 1	Vss	Idd
1 0	Idd	Iss
1 1	Iss	Idd

Una vez que se han agregado fuentes de polarización de voltaje y corriente al circuito, cada fuente de corriente (I) se sintetiza por un espejo de corriente (CM), utilizando el genCM. El tamaño del genCM depende de cuantos tipos de espejo se tengan en una librería, en este trabajo se

sintetizan cuatro diferentes CM [37], por lo tanto el genCM constará de dos bits (a5a6). En la figura 3.7 se muestran los cuatro tipos de espejos utilizados y en la tabla 3.4 se muestra el código del genCM.

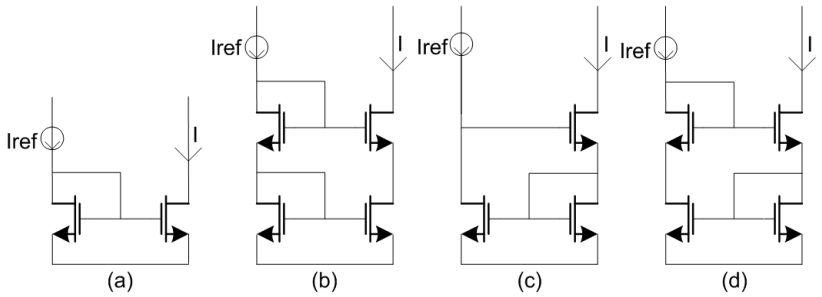

Figura 3.7. Espejos de Corriente. a) Simple, b) Cascode, c) Wilson y d) Wilson modificado.

Tabla 3.4. GenCM: Síntesis de fuentes de corriente por CMs.

a5a6	Espejo de Corriente
00	Simple
01	Cascode
10	Wilson
11	Wilson Modificado

En resumen, para codificar un VF modelado con un nullator a partir de la figura 3.2.a, se tiene un cromosoma (código) de 7 bits: a0..a6, divididos en cuatro diferentes genes: genSS, genSMos, genBias y genCM. Esto significa que existen 2^7=128 diferentes topologías de circuitos que pueden sintetizar un VF con la descripción dada. Sin embargo, algunas topologías no son funcionales, ya que la entrada o la salida puede estar conectada a V_{DD} o a V_{SS}; asimismo, otras topologías no cumplirán con las especificaciones de un VF ($V_O = V_i$).

Para sintetizar un VF de dos MOSFETs a partir de la figura 3.2.b o de cuatro MOSFETs (figura 3.2.c), el cromosoma crece en número de bits pero siempre divididos en cuatro tipos de genes, como se muestra en la tabla 4.5. El genSS se divide en pares de bits, cada par sintetiza un MOSFET como se describe en la tabla 3.1, El genSMos sintetiza un tipo de transistor por cada bit como se describió en párrafos anteriores (tabla 3.2). El genBias se puede dividir en pares y cada par realiza la asignación

de fuentes de polarización según se mostró en la tabla 3.3. El genCM no tiene variación ya que este sintetiza un CM por cada fuente de corriente que existe en el circuito según el código del genBias. El genCM depende de cuantos tipos de CM se quieran sintetizar y estos estén disponibles en una librería.

Tabla 3.5. Número de bits para cada cromosoma, dividido en 4 genes.

VF	genSS	genSMos	genBias	genCM	TOTAL	Figura
1 MOS	2	1	2	2	7 bits	3.2.a
2 MOS	4	2	4	2	12 bits	3.2.b
4 MOS	8	4	8	2	22 bits	3.2.c

La tabla 3.5 muestra que se tienen 2^{12} = 4,096 diferentes topologías posibles para realizar un VF partiendo de la figura 3.2.b (2 nullators). Para sintetizar un VF partiendo de la figura 3.2.c (4 nullators) se tienen 2^{22} = 4,194,304 posibles topologías distintas. De igual manera que un VF modelado con un solo nullator ciertas topologías no tienen sentido ya que la entrada o la salida pueden estar conectadas a V_{DD} o a V_{SS}; y otras posibles topologías no cumplan con las especificaciones de un seguidor de voltaje. Así que ahora se tiene el problema de seleccionar las topologías que funcionan correctamente.

3.3.2 Fenotipo de un VF.

Como se describió en el capítulo 2 el fenotipo de este trabajo es el circuito sintetizado con transistores MOS. En este apartado se sintetizará un cromosoma, partiendo del modelo de un VF de 4 nullators (figura 3.2.c). Se propone el cromosoma 560828, que corresponde en binario a 00100010000111010111100. Consecuentemente, el genSS es 00100010. El modelo de pares O-P que se obtiene, según la tabla 3.1, se muestra en la figura 3.8; donde cada par O-P sintetizará un MOSFET.

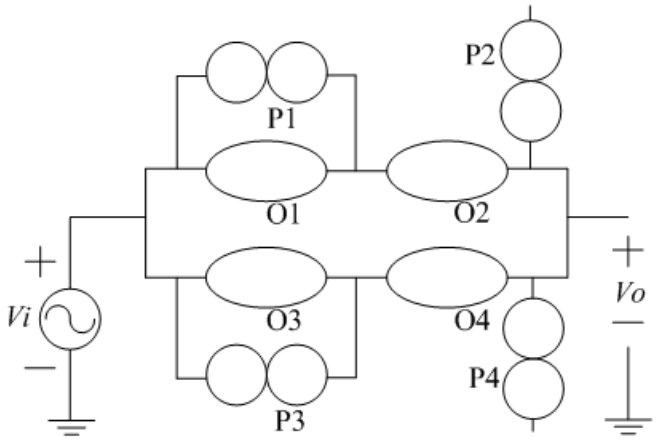

Figura 3.8. Modelo de pares O-P para el genSS=00100010.

Para agregar fuentes de corriente al circuito se respeta la tabla 3.3. Según el cromosoma 560828 el genSMos es 0011 y el genBias es 10101111; por lo tanto para el MOSFET M1 se conecta Idd al drain e Iss al source, de la misma manera M2; para M3 y M4 se conecta Iss al drain e Idd al source. Esto se puede ver en la figura 3.9. Se observa que se pueden eliminar las fuentes de corriente Idd3, Iss1, Idd4 e Iss2 ya que estas no polarizan al circuito.

El genSMos es 0011, significando que los MOSFET sintetizados serán N-M1, N-M2, P-M3 y P-M4 (tabla 3.2), considerando esto se sintetiza el circuito de la figura 3.10, como se puede observar este es el modelo ideal de un VF conocido [2,15,29].

Para que las topologías de los circuitos estén sintetizadas totalmente con transistores, falta sintetizar las fuentes de corriente ideales por espejos de corriente (CMs). Para este ejemplo el genCM es 00, así que se agregaran espejos de corriente simples. Por lo tanto, el circuito final generado es el mostrado en la figura 3.11; este corresponde al cromosoma 560828 (00100010-0011-10101111-00).

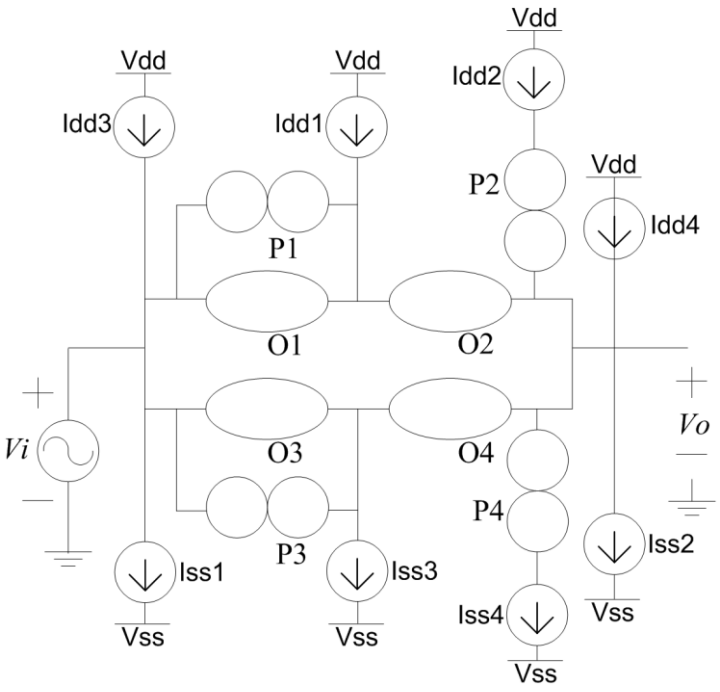

Figura 3.9. Modelo de pares O-P para el genSS=00100010, con polarización del genBias=10101111.

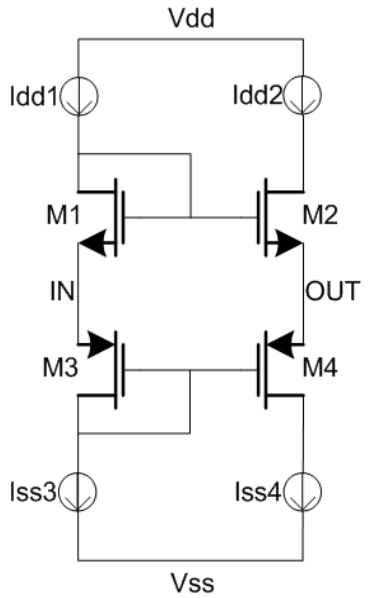

Figura 3.10. Modelo ideal de un VF conocido.

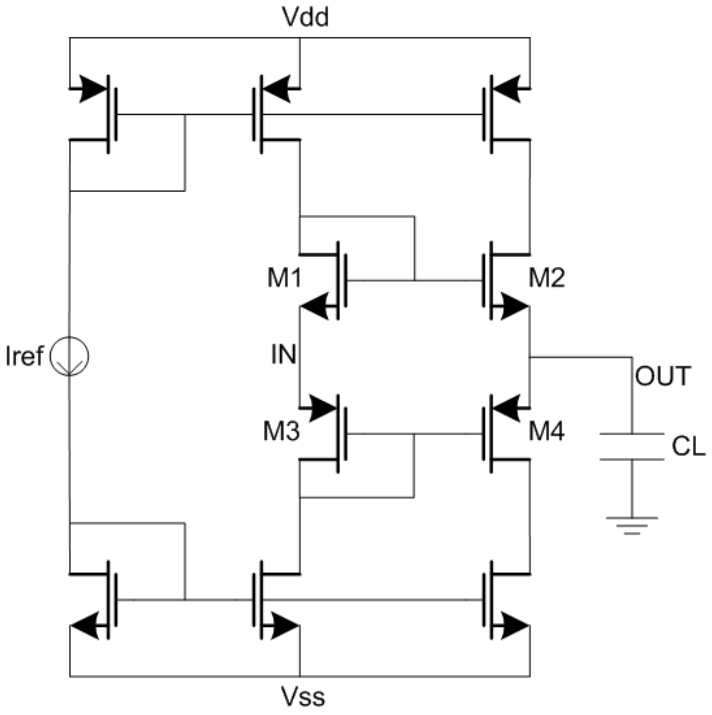

Figura 3.11. Circuito VF MOSFET que corresponde al cromosoma 560828, 00100010-0011-10101111-00.

3.4 Codificación genética de un CF.

3.4.1 Genotipo de un CF.

Partiendo del modelo ideal de un CF utilizando solo un elemento P (figura 3.3.a), se procede a la creación de un circuito CF con transistores MOS. De manera similar a la codificación del VF, cada elemento P debe estar unido a un elemento O para propósito de síntesis del MOSFET. De esta manera, un elemento O puede agregarse a un elemento P a la izquierda (nodo i), a la derecha (nodo j), o en paralelo (entre los nodos i y j); como se ilustra en la figura 3.12. Cada par O-P se puede representar por un gen de 2 bits de longitud; esta codificación crea el gen de pequeña señal (genSS), el cual se describe en la tabla 3.6.

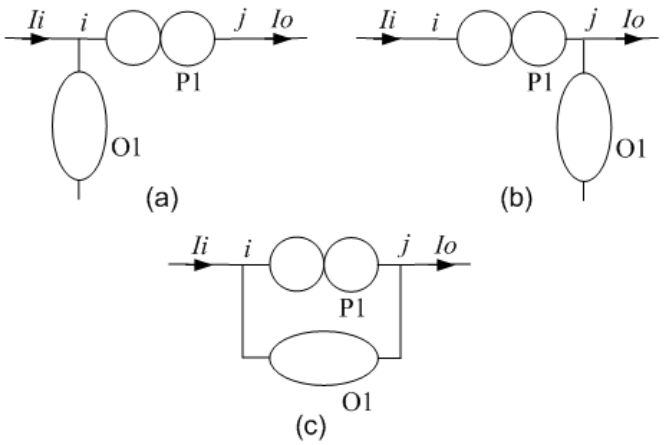

Figura 3.12. Adición de un elemento O: a) En el nodo i, b) en el nodo j y c) entre los nodos i y j.

Tabla 3.6. Codificación del genSS a partir de la figura 3.12.

a0	a1	Unión	
0	0	Pij	Punto de unión i (Figura 3.12.c)
0	1	Pi	(Figura 3.12.a)
1	0	Pj	(Figura 3.12.b)
1	1	Pji	Punto de unión j (Figura 3.12.c)

Cada par O-P se sintetiza por un MOSFET como se mostró en la figura 3.4, en este proceso se utiliza el gen de síntesis del MOSFET (genSMos), el cual es idéntico al descrito para el VF en la tabla 3.2.

El siguiente paso consiste en agregar fuentes de polarización. Primero se utiliza el gen de polarización (genBias); el cual contiene dos bits (a3a4). En todas las combinaciones tanto al drain como al source se les conectaran fuentes de corriente a Vdd y a Vss llamadas Idd e Iss, respectivamente. Segundo a la terminal gate de los MOSFETs se conectara una fuente de voltaje positivo (V+) o negativo (V-); esto se resume en la tabla 3.7.

Tabla 3.7. GenBias: Bits de asignación de fuentes de polarización.

genBias	Conexión		
a3a4	*Drain*	*Source*	*Gate*
0 0	Idd	Iss	V+
0 1	Iss	Idd	V+
1 0	Idd	Iss	V-
1 1	Iss	Idd	V-

Al igual que para los VFs, cada fuente de corriente se sintetiza por un CM, utilizando el genCM. El genCM, como ya se ha mencionado, depende de cuantos tipos de espejos se contengan en una librería. Se tomarán los cuatro tipos de espejos de corriente utilizados para los VFs (figura 3.7) y la tabla 3.4 muestra el código del genCM.

En resumen, para codificar un CF modelado con un solo norator a partir de la figura 3.3.a, se tiene un cromosoma de 7 bits: a0..a6, divididos en cuatro diferentes genes. Sin embargo, algunas topologías no cumplirán con las especificaciones de un CF.

Para sintetizar un CF de dos transistores MOS a partir de la figura 3.3.b o de cuatro MOSFET (figura 3.3.c) el cromosoma crece en número de bits pero siempre divididos en cuatro tipos de genes, como se muestra en la tabla 3.8. El genSS se divide en pares de bits, cada par sintetiza un MOSFET como se describe en la tabla 3.6, El genSMos sintetiza un tipo de MOS por cada bit como se describe en la tabla 3.2. El genBias se puede dividir en pares y cada par realiza la asignación de fuentes de polarización según se muestra en la tabla 3.7. El genCM no tiene variación ya que este sintetiza un CM por cada I que existe en el circuito según el código del genBias.

Tabla 3.8. Número de bits para cada cromosoma, dividido en 4 genes.

CF	genSS	genSMos	genBias	genCM	TOTAL	Figura
1 MOS	2	1	2	2	7 bits	3.3.a
2 MOS	4	2	4	2	12 bits	3.3.b
4 MOS	8	4	8	2	22 bits	3.3.c

De igual manera que un VF modelado con un nullator ciertas topologías no son prácticas para los CFs, ya que no cumplirán con las especificaciones de un seguidor de corriente. Por lo tanto, el problema será detectar las topologías que tengan un funcionamiento correspondiente al de un CF y esto se realizara por medio de simulaciones en SPICE. La compuerta de los MOSFET siempre se conectara a una fuente de voltaje con el fin de que el transistor trabaje en la región de saturación.

3.4.2 Fenotipo de un CF.

Se sintetiza el cromosoma 838716 partiendo del modelo de un CF de 4 norators (figura 3.3.c). El cromosoma en binario es 00110011-0011-00001111-00; por lo tanto, el genSS es 00110011, el genSMos es 0011, el genBias es 00001111 y el genCM es 00. El modelo de pares O-P que se obtiene con el genSS y la tabla 3.6 se tiene la polarización mostrada en la figura 3.13; donde se han eliminando las fuentes de corriente que no polarizan al circuito ya que tienen una trayectoria directa de Vdd a Vss.

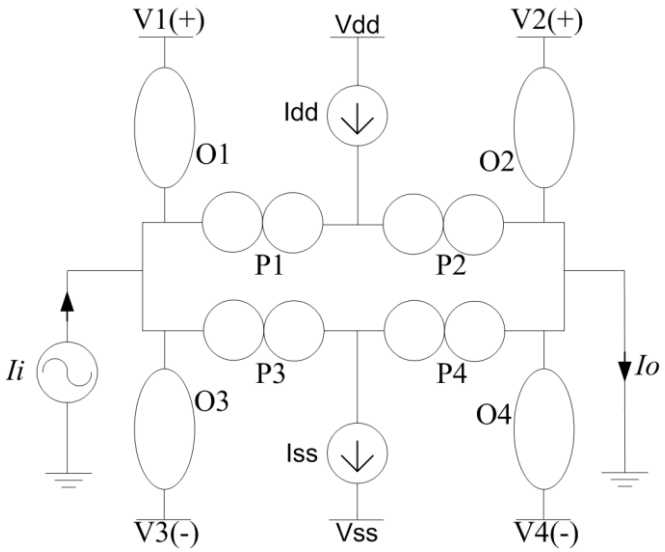

Figura 3.13. Modelo de pares O-P para el genSS=00110011 con polarización del genBias=00001111.

Siendo el genBias 00001111; los transistores M1 y M2 se conectan por su terminal drain a Idd y por el source se conectan a Iss; la terminal gate de ambos es conectada a una fuente de voltaje positiva (V1 y V2) y los transistores M3 y M4 se les conecta una fuente Iss al drain y una fuente Idd a la terminal source, mientras la terminal gate de estos es conectada a una fuente de voltaje negativa, en la figura 3.13 se muestran por V3 y V4.

Según el cromosoma 838716 el genSMos es 0011, significando que los MOSFET sintetizados serán N-M1, N-M2, P-M3 y P-M4 (tabla 3.2). Entonces, se sintetiza el circuito de la figura 3.14 el cual resulta de la

figura 3.13, como se puede observar este es un modelo ideal de un CF conocido [2,29].

Para que la topología esté diseñada totalmente con transistores MOS, falta sintetizar las fuentes de corriente ideales por CMs. Para este ejemplo, el genCM es 00 por lo tanto se sintetizan por espejos de corriente simples las fuentes Idd e Iss. Por lo tanto, el circuito final generado por el cromosoma 838716 (00110011001100000111100), es el mostrado en la figura 3.15.

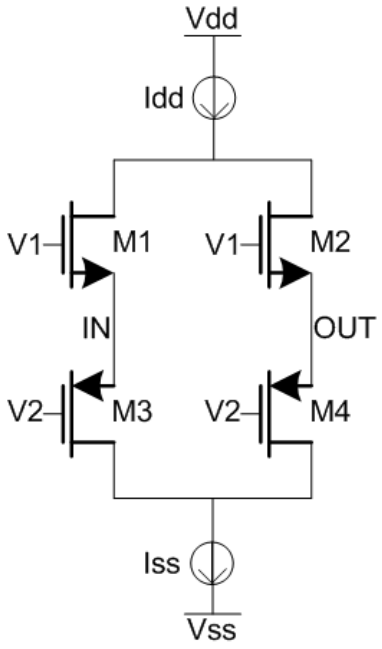

Figura 3.14. Modelo ideal de un CF conocido.

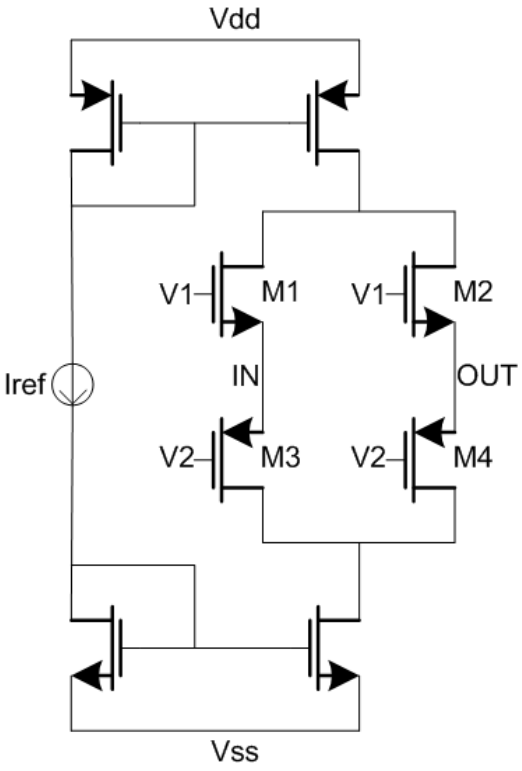

Figura 3.15. Circuito CF MOS que corresponde al cromosoma 838716, 00110011001100000111100.

3.5 Codificación genética de un VM.

3.5.1 Genotipo de un VM.

Un circuito espejo de voltaje (VM) es realizado conectando a la entrada de un VF dos MOSFETs como se muestra en la figura 3.16 [16,38]. Por lo tanto, el cromosoma propuesto del VM es muy parecido al del VF, solo se incrementa en un gen de un bit para diferenciar entre un inversor de entrada P o entrada N (genInv), esto se detalla en la tabla 3.9. En la tabla 3.10 se muestra el número de bits para cada cromosoma de los VMs dividido en 5 genes; los cuatro primeros idénticos a los de un VF y el quinto llamado genInv.

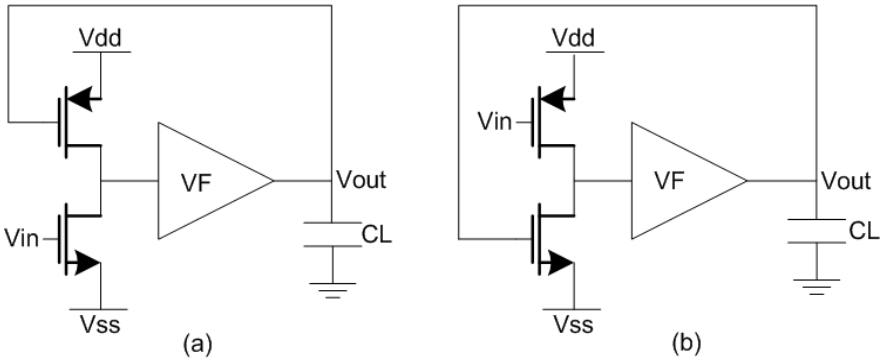

Figura 3.16. Espejo de voltaje (a) tipo N, (b) tipo P.

Tabla 3.9. GenInv: Adición del bloque de inversión.

genInv	Conexión
a7	
0	Tipo P
1	Tipo N

Tabla 3.10. Número de bits para cada cromosoma, dividido en 5 genes.

VM	genSS	genSM	genBias	genCM	genInv	TOTAL	Figura
1 MOS	2	1	2	2	1	8 bits	3.2.a
2 MOS	4	2	4	2	1	13 bits	3.2.b
4 MOS	8	4	8	2	1	23 bits	3.2.c

3.5.2 Fenotipo de un VM.

Partiendo del ejemplo del fenotipo de un VF mostrado en párrafos anteriores, se propone para esta sección encontrar el circuito equivalente del cromosoma 00100010001110101111000 de un VM. Si a este cromosoma se le elimina el último bit, se puede observar que se convierte al cromosoma mostrado en el ejemplo de un VF. Utilizando el genInv igual a 0 y la figura 3.10, que muestra la síntesis del cromosoma VF; se sintetiza el circuito de la figura 3.17, como se puede observar este es un modelo ideal de un VF conocido [16,38].

Al sintetizar las fuentes de corriente ideales por espejos de corriente simples debido a que en este caso el genCM es 00, se obtiene el circuito final del VM que le corresponde el cromosoma 00100010-0011-10101111-00-0 (figura 3.18).

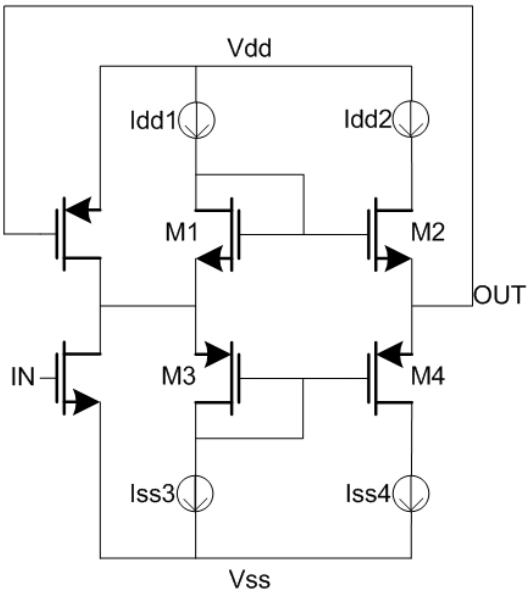

Figura 3.17. Modelo ideal de un VM conocido.

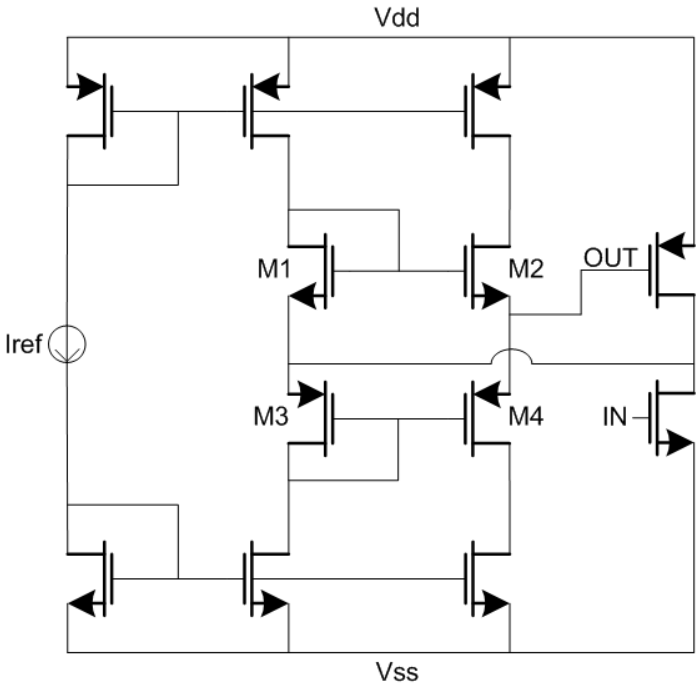

Figura 3.18. Circuito VM MOS que corresponde al cromosoma 1121656, 00100010-0011-10101111-00-0.

3.6 Codificación genética de un CM.

3.6.1 Genotipo de un CM.

Los espejos de corriente son bloques básicos de polarización y pueden considerarse como un caso particular de los CFs. En la figura 3.19 se

utilizan celdas de dos, tres y cuatro norators para formar los espejos de corriente. En estas celdas se tiene un punto de referencia, este puede considerarse como tierra o un voltaje de polarización positivo o negativo. Las variables eléctricas para estas celdas son i_i e i_o. Los nodos IN, OUT, A y B se utilizaran para agregar nullators y/o fuentes polarización [36,39].

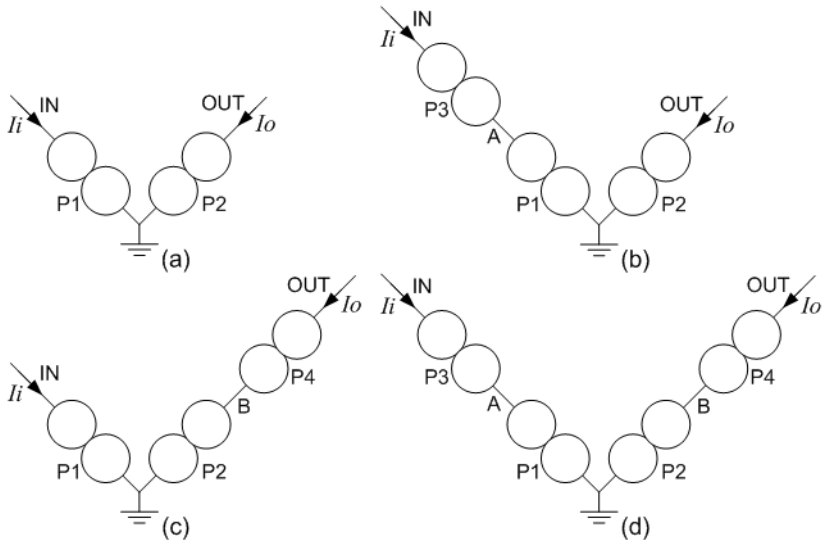

Figura 3.19. Celdas genéricas para formar espejos de corriente.

Partiendo del modelo ideal de un CM utilizando cuatro elementos P (figura 3.19.d), se procede a la creación de un circuito CM con MOSFET. Cada elemento P debe estar unido a un elemento O y puede agregarse a la izquierda, a la derecha o en paralelo; como se ilustró en la figura 3.12. Cada par O-P se puede representar por un gen de 8 bits de longitud (a0…a7); esta codificación crea el gen de pequeña señal (genSS), el cual se propone idéntico al utilizado para un CF y se muestra en la tabla 3.11. Cada par O-P se sintetiza por un MOSFET, se asume que el punto de unión del CM ideal es GND; el tipo de MOSFET que debe sintetizarse según el genSS se muestra en la tabla 3.11. Esto para asegurar que estarán en saturación los MOSFET, por lo tanto en este caso no es necesario un gen de síntesis del MOSFET como en los anteriores cromosomas.

Tabla 3.11. Codificación del genSS para un CM.

a0	a1	Unión		Tipo MOS
0	0	Pij	Punto de unión i (Figura 3.12.c)	P-MOS
0	1	Pi	(Figura 3.12.a)	N-MOS
1	0	Pj	(Figura 3.12.b)	N-MOS
1	1	Pji	Punto de unión j (Figura 3.12.c)	N-MOS

En todas las combinaciones posibles los nodos IN y OUT se conectaran fuentes de corriente a Vdd (Idd) para la polarización de los cuatro MOSFET. La terminal gate de los transistores se le conectara a los nodos IN, OUT, A o B y en caso de que no se conecte a ningún nodo la terminal gate se conectará a una fuente de voltaje positiva (V+). Se describe esto utilizando un gen de polarización (genBias); este gen contiene seis bits (a8…a13) y en las tablas 3.12 se describe la decodificación de cómo conectar la terminal gate de cada MOSFET.

Tabla 3.12. GenBias: Bits de polarización del circuito.

a8a9	gate M1
00	V+
01	B
10	IN
11	OUT

a10a11	gate M2
00	V+
01	A
10	IN
11	OUT

a12	gate M3
0	V+
1	OUT

a13	gate M4
0	V+
1	IN

En resumen, para codificar un CM modelado con cuatro norators a partir de la figura 3.19.d, se tiene un cromosoma de 14 bits: a0..a13, divididos en dos diferentes genes. Para sintetizar un CM de dos o tres MOSFETs a partir de la figuras 3.19.a, 3.19.b y 3.19.c; el cromosoma decrece en número de bits pero siempre divididos en dos tipos de genes, como se muestra en la tabla 3.13. El genSS se divide en pares de bits, cada par sintetiza un MOSFET como se describe en la tabla 3.6, El genBias se puede dividir en cadenas de uno o dos bits y cada cadena realiza la asignación de fuentes de polarización según se muestra en la tabla 3.12.

Tabla 3.13. Número de bits para cada cromosoma, dividido en 2 genes.

CM	genSS	genBias	TOTAL	Figura
2 MOS	4 bits	a4-a5	6 bits	3.19.a
3 MOS	6 bits	a6a7-a8a9-a10	11 bits	3.19.b
3 MOS	6 bits	a6a7-a8a9-a10	11 bits	3.19.c
4 MOS	8 bits	a8a9-a10a11-a12-a13	14 bits	3.19.d

3.6.2 Fenotipo de un CM.

Partiendo del modelo de un CM de 4 norators (figura 3.19.d), Se sintetizará el cromosoma 11011101000101 (14149). En la figura 3.20 se muestra la descripción del genSS= 11-01-11-01, considerando la descripción de la tabla 3.11.

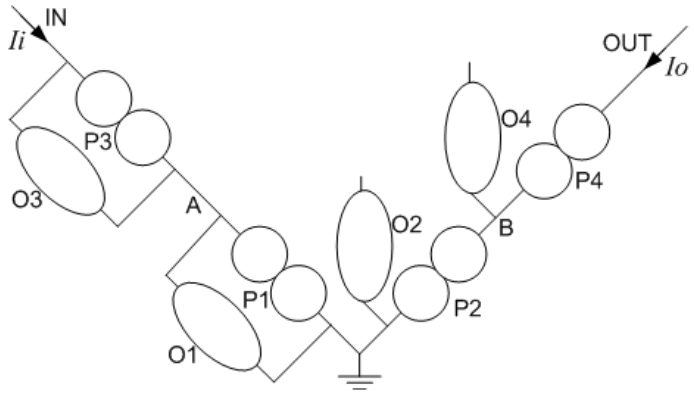

Figura 3.20. Modelo de pares O-P para el genSS=11011101.

El genBias en este ejemplo es 00-01-0-1, utilizando la tabla 3.12 el gate del transistor M1 se conecta a una fuente positiva (a8a9=00), sin embargo como el gate ha sido conectado al nodo A, esta fuente no se sintetiza, como se explicó en la sección del genotipo CM. a10a11=01 significan que la terminal gate del transistor M2 es conectada al nodo A; a12=0 en la tabla 3.12 significa que la terminal gate de M3 se conecta a una fuente positiva, la cual no es sintetizada ya que esta terminal se ha conectado al nodo IN. Por último, a13=1 en la tabla 3.12 significa que la terminal gate de M4 se conecta al nodo IN. Realizando las conexiones correspondientes se obtiene la figura 3.21.

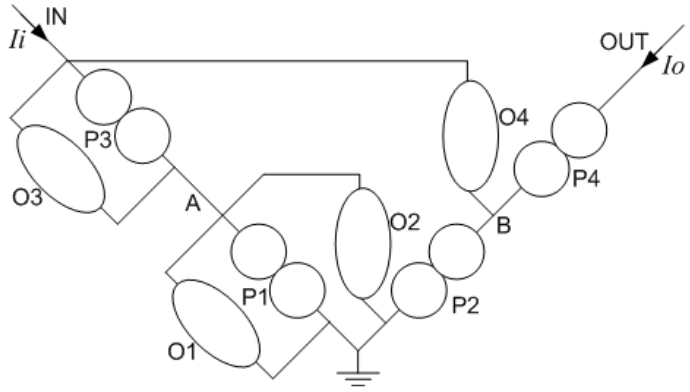

Figura 3.21. Modelo de pares O-P para el genSS=11011101,
y polarización del genBias=000101.

La figura 3.21 muestra el modelo en pares O-P del CM que es sintetizado por transistores en la figura 3.22; como puede observarse es un CM conocido como espejo cascode [16,36,37].

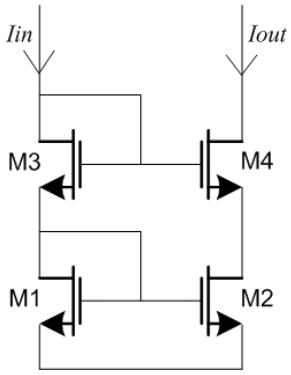

Figura 3.22. CM sintetizado con el cromosoma 14149, espejo cascode.

3.7 Diagrama de flujo del método de síntesis propuesto.

En el capítulo 2 se mostró la estructura general de los algoritmos genéticos (figura 2.1). Tomando como base ese diagrama general se muestra el GA propuesto en la figura 3.23. Donde, primero se crean aleatoriamente algunos individuos (generación espontánea) para generar una población inicial (población 0). Esta población inicial se hace pasar por una primera selección, la cual consiste en discriminar los códigos de los circuitos que no son válidos; ya que no todos los códigos sintetizan topologías válidas, como se explicó en el capítulo 2 [3-5,14,32-34]. Si no

se selecciona al menos un individuo que sea sintetizable, se procede a crear una nueva población nuevamente por generación espontánea.

Una vez que al menos un individuo sea una topología válida se crean los archivos F1, donde se sintetiza el circuito según su cromosoma, se agrega un archivo que contiene la tecnología de CIs (apéndice B) y se polariza el VF con fuentes ideales. Posteriormente, los archivos F1 son ejecutados en SPICE y este genera un archivo de salida tipo *.txt.

El programa toma el archivo de salida de SPICE y evalúa Vo>A*Vi, donde A es una variable de aptitud que se puede modificar. Esta variable puede tener un valor entre 0.7 y 1, el diseñador elige el valor según sus necesidades. Una segunda selección desecha todas las topologías que no funcionan como VFs, si la actual población no cuenta con al menos un individuo que sea VF, se procede a crear una nueva población por generación espontánea.

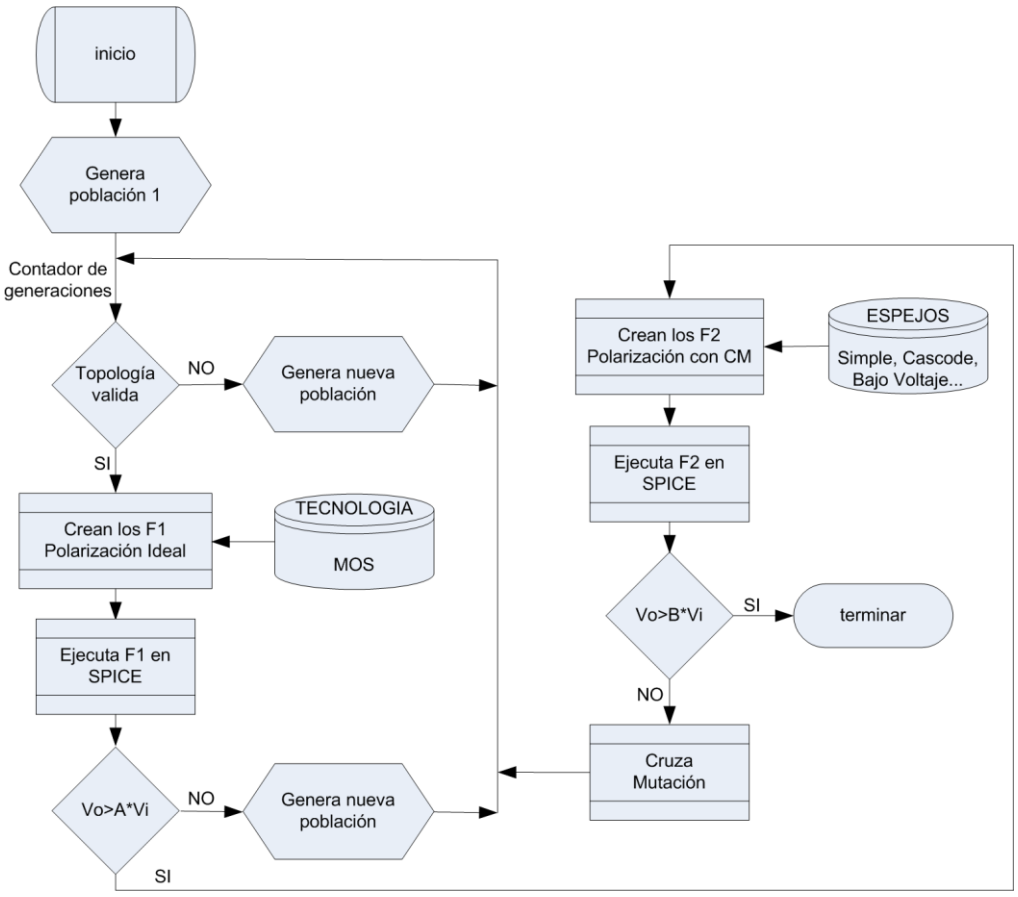

Figura 3.23. Diagrama de flujo para el GA que sintetiza VFs.

Una vez que al menos un individuo se comporte como VF, se crean los archivos F2. En el archivo F2 se sintetizan las fuentes de corriente ideales por espejos de corriente según el genCM del cromosoma y los archivos F2 son ejecutados en SPICE.

El programa toma el archivo de salida de SPICE y evalúa Vo>B*Vi, donde B es una variable de aptitud que se puede modificar. Esta variable es similar a 'A', el diseñador elige el valor según sus necesidades. Esta tercera selección evalúa si el VF cumple con las especificaciones pedidas por el usuario; si es así el programa ha generado una solución y es detenido. Si ninguno de los circuitos realizados cumple con las especificaciones se procede a generar otra población, partiendo de las operaciones de cruza y mutación descritas en el capítulo 2.

Por último, el ciclo se repite hasta que se llegue a una solución o a un número máximo de generaciones para encontrar la solución.

El diagrama de flujo propuesto en la figura 3.23 funciona para los tres casos de VFs mostrados en la figura 3.2. Conjuntamente, como el genotipo de los VMs esta formado por un cromosoma idéntico al de los VF más un genInv, el diagrama de flujo de la figura 3.23 describe como realizar el algoritmo genético para encontrar un VM, según los casos de la figura 3.16. Primero se crea una población inicial (aleatoriamente), después se verifica que existan en la población la topologías válidas y se generan archivos F1; donde se sintetiza el circuito VM según su cromosoma polarizado con fuentes ideales. El programa simula en SPICE el circuito y evalúa Vo>A*Vi, (donde A es la variable de aptitud), si la actual población no cuenta con al menos un individuo que sea considerado VM, se procede a crear una nueva población por generación espontánea.

Una vez que al menos un individuo se comporte como VM, se crean los archivos F2, donde se sintetizan las fuentes de corriente ideales por CMs y los archivos F2 son ejecutados en SPICE. Se evalúa Vo>B*Vi, (donde B es la segunda variable de aptitud), si el circuito con transistores cumple

con las especificaciones pedidas por el usuario el programa ha generado una solución y es detenido. Si ninguno de los circuitos realizados cumple con las especificaciones se procede a generar otra población, partiendo de las operaciones de cruza y mutación. El ciclo se repite hasta que se encuentre a un circuito VM que cumpla con la especificación B.

Para desarrollar un GA que encuentre topologías de CFs sólo basta con modificar en el diagrama 3.23 las medidas de aptitud del circuito; cambiando Vo>A*Vi por Io>A*Ii y Vo>B*Vi por Io>B*Ii. Realizando los cambios, el diagrama de flujo del GA que genera CFs se muestra en la figura 3.24; donde también se nota que no es necesario realizar la evaluación de topología válida; puesto que, por la descripción del cromosoma todas las topologías de CFs serán válidas.

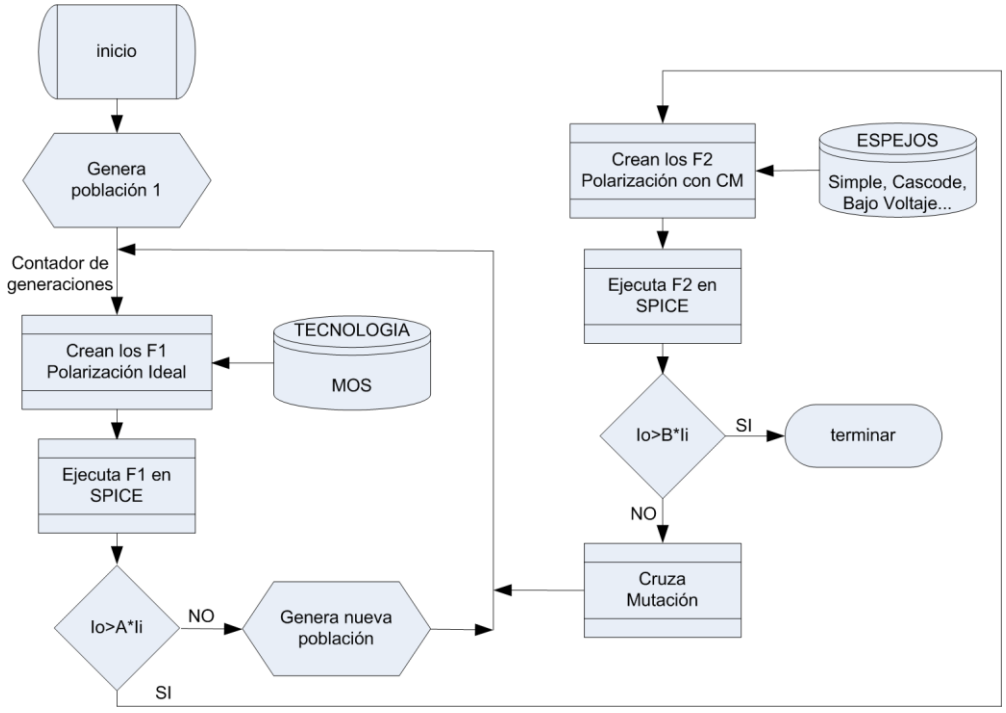

Figura 3.24. Diagrama de flujo del GA que sintetiza CFs.

En el diagrama de flujo para sintetizar espejos de corriente, también se debe de eliminar la evaluación de topología válida; puesto que, por la descripción del cromosoma todas las topologías de CMs serán validas. Así mismo no es necesario realizar dos evaluaciones del circuito ya que

su cromosoma no cuenta con un genCM. Así, el diagrama de flujo propuesto para el GA que encuentra topologías de CM se muestra en la figura 3.25.

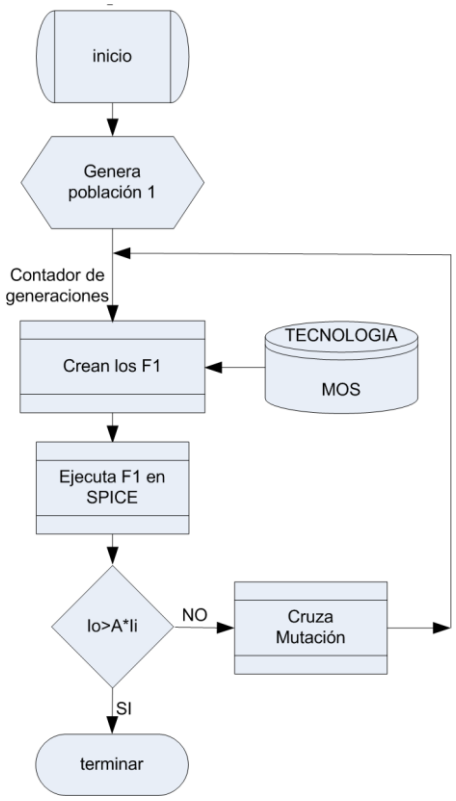

Figura 3.25. Diagrama de flujo del GA que sintetiza CMs.

Los diagramas de flujo mostrados en las figuras 3.23 (para VFs y VMs), 3.24 (para CFs) y 3.25 (para CMs) se han programado en MatLab. En el apéndice B se muestra una descripción de los archivos F1 y F2 generados por MatLab para su simulación en SPICE. Se muestran algunas topologías obtenidas de estos GAs en el capítulo 4.

Capítulo 4

Síntesis y evolución de bloques de ganancia unitaria.

4.1 Introducción.

El algoritmo genérico propuesto para la síntesis de bloques de ganancia unitaria (descrito en 3.7) se programo en MatLab. En este capítulo se muestran algunas topologías obtenidas con la herramienta desarrollada. Las topologías de VFs encontradas se muestran en la sección 4.2, donde se muestra en forma detallada como se ejecuta el programa y como este encuentra una topología. Los circuitos de CFs, VMs y CMs encontrados se muestran en las secciones 4.3 a 4.5, ya en estas secciones no se describe tan detalladamente el funcionamiento del programa realizado, debido a que son similares a la sección 4.2. Finalmente, se describe como los cromosomas propuestos pueden evolucionar para convertirse en topologías más complejas como CCs.

4.2 Topologías de VFs generadas por el GA.

Como se menciono en la introducción, el GA se programó en MatLab y en este apartado se muestran algunas topologías obtenidas por la ejecución de éste. Para que MatLab pueda ejecutar los programas realizados su directorio debe estar situado en la carpeta actual de trabajo; la figura 4.1 muestra la carpeta de trabajo principal de MatLab, dentro de esta se han creado 13 carpetas (una por cada programa realizado). Dentro de cada carpeta se encuentran subprogramas y un programa principal que encuentra las topologías de los bloques de ganancia unitaria. En la figura 4.2 se muestra la carpeta del programa que genera los CFs de un

MOSFET; como se ve, dentro de esta carpeta se encuentra el ejecutable de T-SPICE y la tecnología CMOS a utilizar, en este trabajo de 0.35µm (mos35t.Lib, apéndice B).

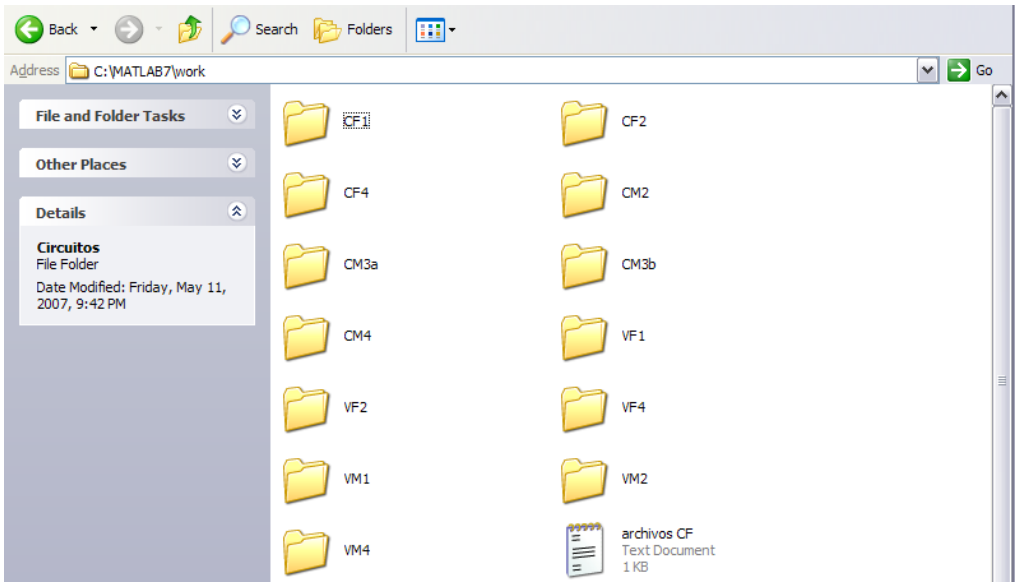

Figura 4.1. Carpeta work de MatLab.

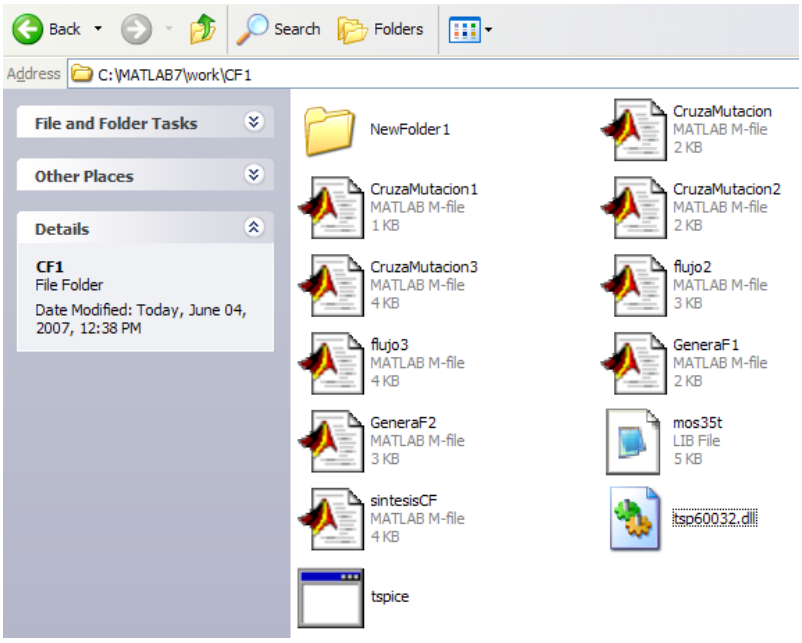

Figura 4.2. Carpeta del programa que genera los CFs de 1 MOSFET.

En la figura 4.2 se observa una carpeta llamada NewFolder1, al ejecutar el programa los archivos creados por MatLab y SPICE se guardan en

esta carpeta. La figura 4.3 muestra la carpeta NewFolder1 del programa que genera CFs de 1 MOSFET; los archivos SP que utiliza SPICE fueron creados por MatLab, mientras que los archivos de texto fueron creados por SPICE al realizar la simulación de los circuitos.

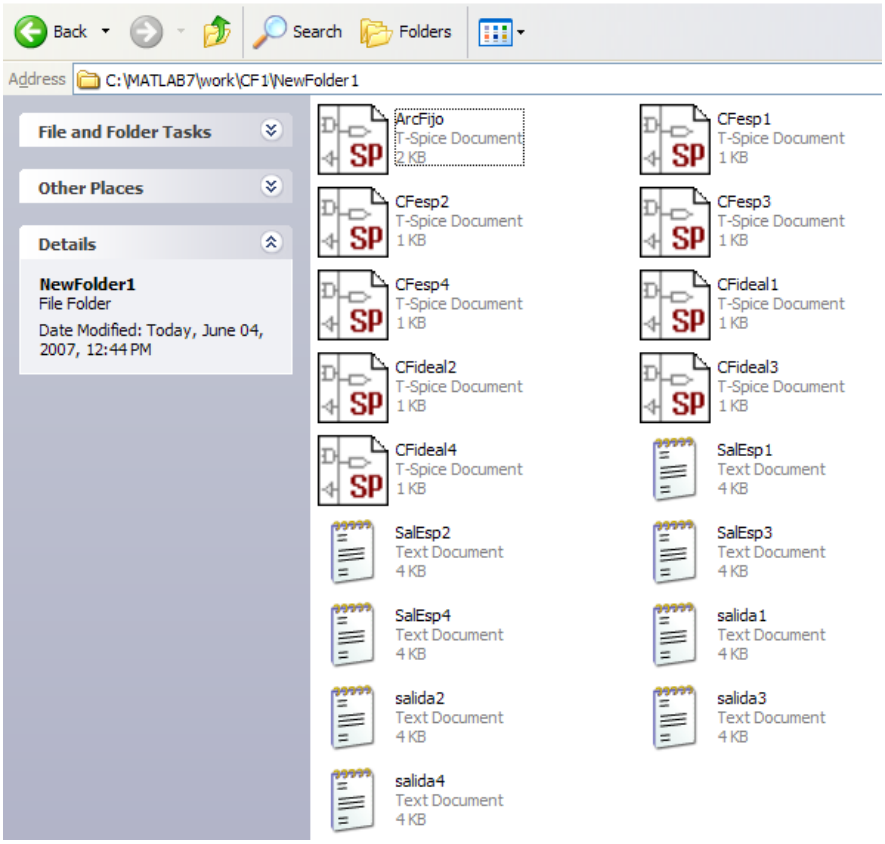

Figura 4.3. Carpeta NewFolder1 del programa que genera los CFs de 1 MOSFET.

Para ejecutar el programa que encuentra las topologías de VFs solo se teclea en MatLab sintesisVF; y este comienza generando una población inicial de 8, 12 ó 20 individuos según se trate de un VF de 1, 2 ó 4 MOSFETs, respectivamente. El tamaño de la población trata de conservarse en las siguientes generaciones; sin embargo puede decrecer hasta un mínimo de 4 individuos.

En el listado 4.1 se muestra una ejecución de 'sintesisVF' de 4 MOSFET, se observa que la generación inicial es de 20 individuos y como el tamaño de la población va disminuyendo en las siguientes generaciones. La evolución del tamaño de la población se observa en la figura 4.4. En el

listado 4.1 también se observa que algunas topologías sintetizadas en MatLab no pueden ser simuladas por SPICE ya que no convergen a un punto de operación en este último. Después de evaluar todos los individuos de una generación, MatLab despliega una tabla donde muestra el número del cromosoma, si éste es sintetizable, si es un VF con fuentes de corriente ideales, si es un VF sustituyendo las fuentes de corriente ideales con CMs.

Listado 4.1. Listado desplegado por MatLab al ejecutar 'sintesisVF' (de 4 MOSFET).

```
sintesisVF
GeneracionF0 =
  Columns 1 through 5
     4043589     2701086      324396     2210020     4144937
  Columns 6 through 10
     1203575     2554327     2450360     3995114     1803573
  Columns 11 through 15
      195765     4102388     1522773     3664082     1725362
  Columns 16 through 20
     1095518     3895158      654006     2702405     1697489

SPICE NO PUEDE SIMULAR
error =
     2701086

SPICE NO PUEDE SIMULAR
error =
     2554327

Cromosoma_EsRealizable_EsVF_EsVFconEsp =

     4043589           1           0           0
     2701086           0           0           0
      324396           1           1           0
     2210020           1           0           0
     4144937           0           0           0
     1203575           0           0           0
     2554327           1           0           0
     2450360           1           1           0
     3995114           1           0           0
     1803573           1           1           0
      195765           1           0           0
     4102388           0           0           0
     1522773           0           0           0
     3664082           1           0           0
     1725362           1           0           0
     1095518           1           0           0
     3895158           0           0           0
      654006           1           0           0
     2702405           1           0           0
     1697489           1           0           0

NINGUN INDIVIDUO DE LA POBLACION ES VF CON ESPEJOS DE CORRIENTE
```

```
se realiza una nueva poblacion a partir de CRUZA Y MUTACION

GeneracionF1 =
  Columns 1 through 5
       62252      308012      320300      848684     2450364
  Columns 6 through 10
     2450361     2450344     2581432     1803581     1803701
  Columns 11 through 12
     1805621     1803569

SPICE NO PUEDE SIMULAR
error =
      320300

SPICE NO PUEDE SIMULAR
error =
     2450344

Cromosoma_EsRealizable_EsVF_EsVFconEsp =

       62252           1           0           0
      308012           1           0           0
      320300           0           0           0
      848684           1           1           0
     2450364           1           1           0
     2450361           1           1           0
     2450344           0           0           0
     2581432           0           0           0
     1803581           1           0           0
     1803701           1           1           0
     1805621           1           0           0
     1803569           1           0           0

NINGUN INDIVIDUO DE LA POBLACION ES VF CON ESPEJOS DE CORRIENTE
se realiza una nueva poblacion a partir de CRUZA Y MUTACION
GeneracionF2 =
  Columns 1 through 5
      910630     3006758     2999590     3007778     2406513
  Columns 6 through 8
     2144373     2406516     2406501

Cromosoma_EsRealizable_EsVF_EsVFconEsp =

      910630           1           0           0
     3006758           1           0           0
     2999590           0           0           0
     3007778           1           0           0
     2406513           1           1           0
     2144373           1           0           0
     2406516           1           1           0
     2406501           1           0           0

NINGUN INDIVIDUO DE LA POBLACION ES VF CON ESPEJOS DE CORRIENTE
se realiza una nueva poblacion a partir de CRUZA Y MUTACION
GeneracionF3 =
  Columns 1 through 5
     2456560      850912     2446048      850656     1809221
  Columns 6 through 8
     2456481     1805125     1809377
```

```
SPICE NO PUEDE SIMULAR
error =
     1809377

Cromosoma_EsRealizable_EsVF_EsVFconEsp =

     2456560           1           0           0
     2417280           1           1           0
     2446048           1           0           0
      850656           1           0           0
     1809221           1           0           0
     2456481           1           0           0
     1805125           1           0           0
     1809377           0           0           0

NINGUN INDIVIDUO DE LA POBLACION ES VF CON ESPEJOS DE CORRIENTE
se realiza una nueva poblacion a partir de CRUZA Y MUTACION
GeneracionF4 =
      2400896      320128       2419328     2417288

Cromosoma_EsRealizable_EsVF_EsVFconEsp =

     2400896           1           1           1
      320128           0           0           0
     2419328           0           0           0
     2417288           1           0           0
```

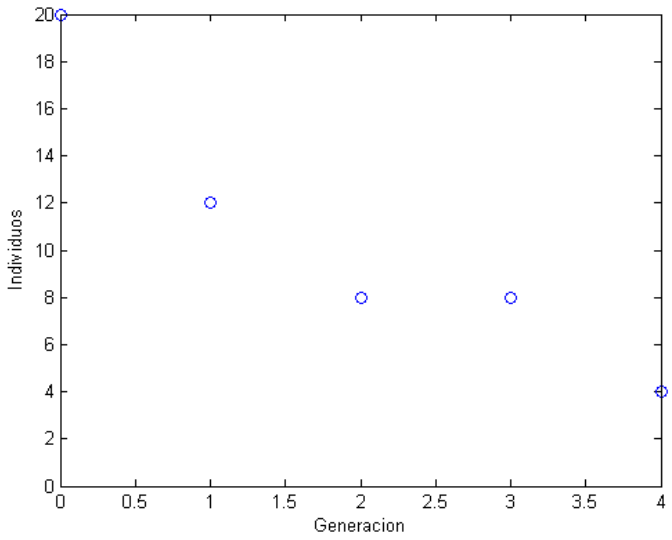

Figura 4.4. Tamaño de la población en las diferentes generaciones.

En la figura 4.5 se muestra el tiempo de ejecución del programa en segundos; fue utilizada una computadora marca eMachines modelo M2352, con procesador Mobile AMD Athlon(tm) XP 3000+ a 524 MHz, con 448MB en RAM. En la figura 4.6 se observa el resultado de la evaluación del mejor de los individuos de cada generación; donde la

aptitud 1 es la evaluación del circuito con fuentes de corriente ideales y la aptitud 2 es la evaluación del circuito con CMs. Cabe mencionar que el programa es detenido cuando algún individuo rebasa la 'aptitud 2' pedida. Para este ejemplo se ejecuto el programa con Vo= 0.8*Vi para la primer aptitud y Vo= 0.7*Vi para la aptitud 2.

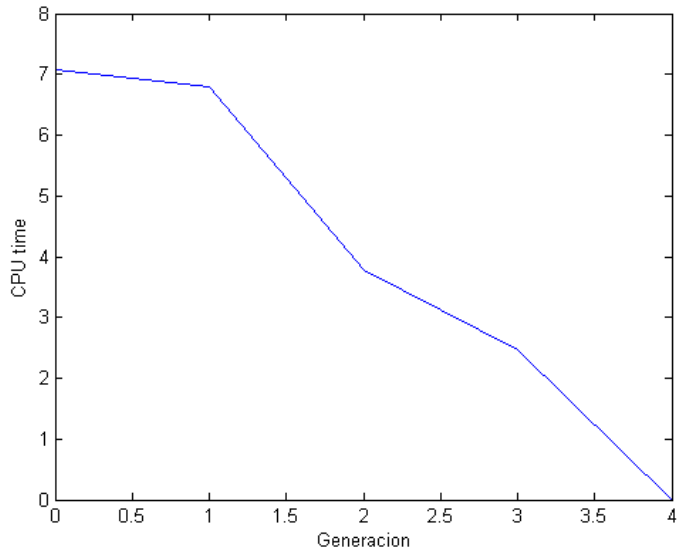

Figura 4.5. Tiempo de ejecución del programa VF de 4 MOSFET.

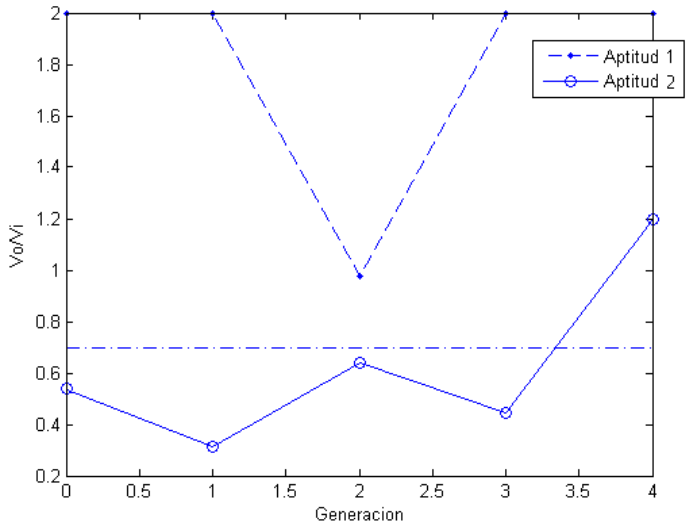

Figura 4.6. Medida de aptitud de los individuos en las diferentes generaciones.

En la figura 4.7 se muestran 4 circuitos que se comportaron como buenos VFs de 1 MOSFET (a partir de la figura 3.2(a)); se observa que el bit más significativo del cromosoma esta a la izquierda y menos significativo a la derecha. Cabe recordar que el criterio para decidir si la topología es un buen circuito VF, se basa solo en su respuesta de ganancia en CA. Por lo tanto, se considera como buen VF aquel que supere la medida de aptitud que el usuario proponga. Así, MatLab podría considerar como VF a un circuito que en realidad no es practico; esto se debe a que solo esta decidiendo según la simulación en SPICE de un sólo parámetro en CA y no en varios. Para asegurar que el circuito sintetizado es práctico, es necesario desarrollar un procedimiento de optimización, lo cual es considerado como un tema de investigación abierto [3-5].

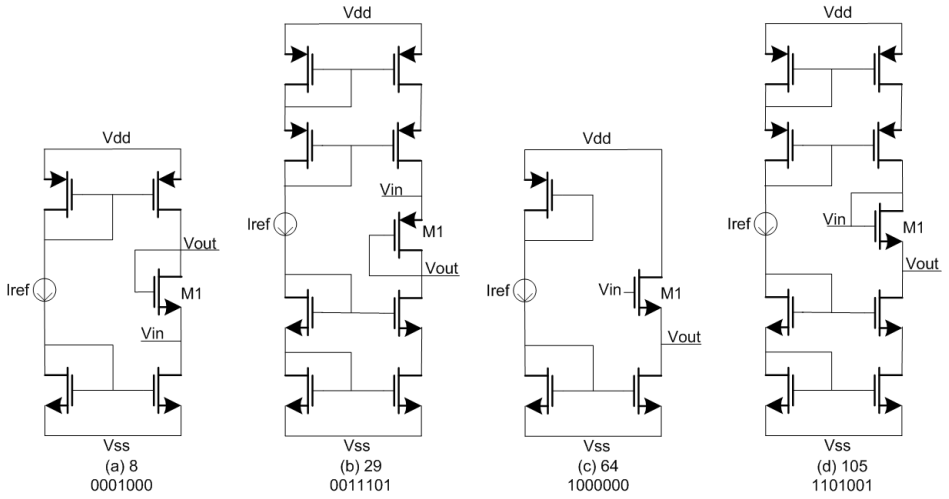

Figura 4.7. Circuitos que se comportaron como VFs de 1 MOSFET.

En la figura 4.8 se observan 4 circuitos que se comportaron como buenos VFs de 2 MOSFETs (a partir de la figura 3.2(b)); y en la figura 4.9 se exponen 6 circuitos que se comportaron como VFs de 4 MOSFETs (a partir de la figura 3.2(c)). En la figura 4.10 se muestra la ganancia de voltaje de los circuitos de la figura 4.9 (a) 110381, (b) 1369000, (c) 1633983, (d) 1715676, (e) 2959967 y (f) 3711904.

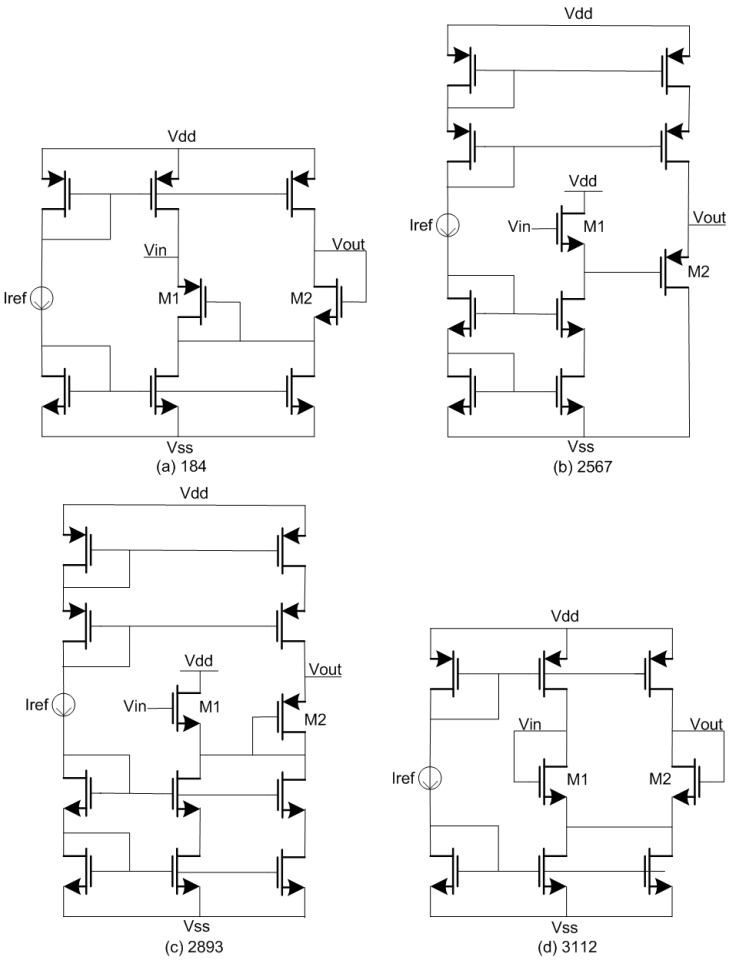

Figura 4.8. Circuitos que se comportaron como VFs de 2 MOSFETs.

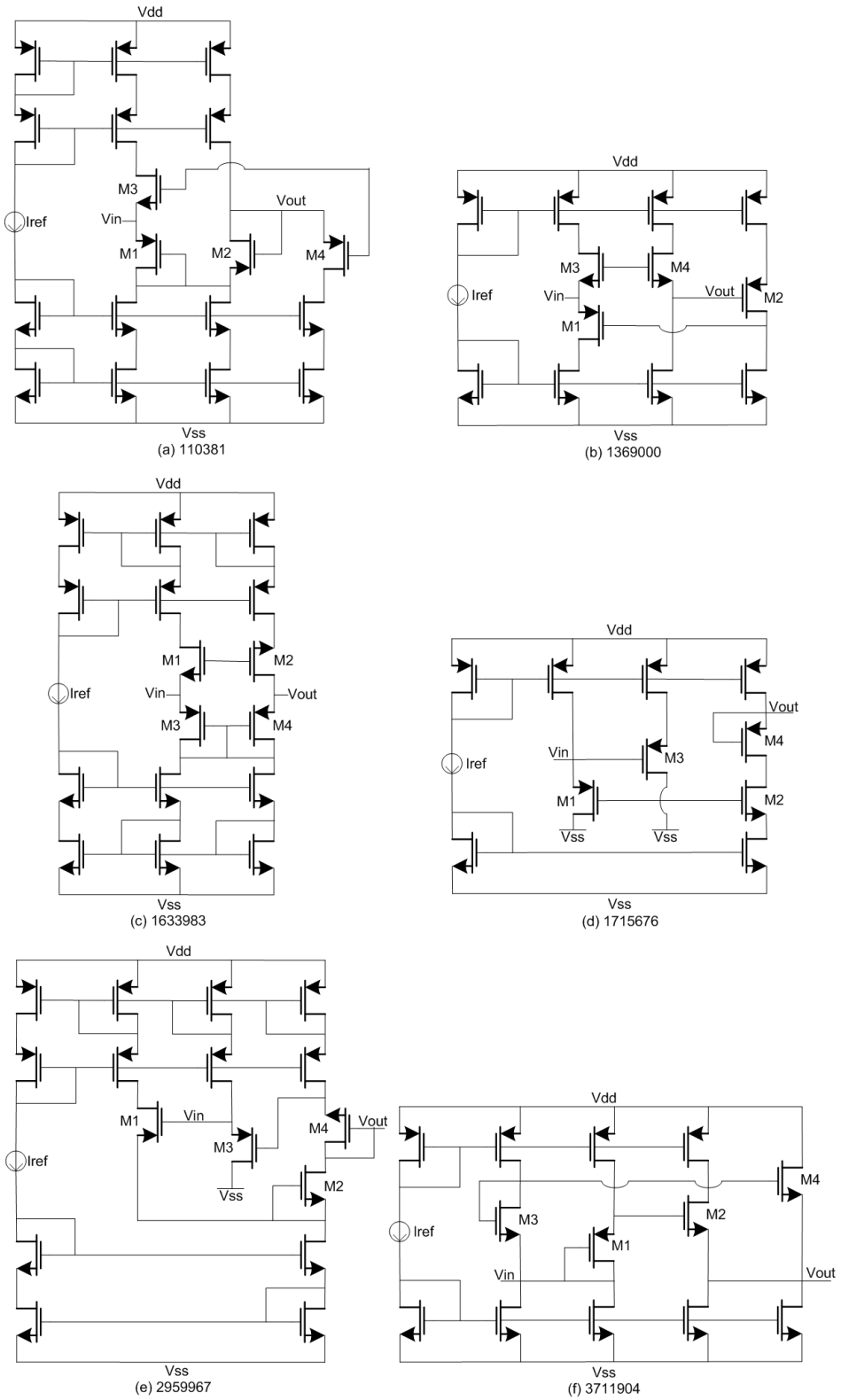

Figura 4.9. Circuitos que se comportaron como VFs de 4 MOSFETs.

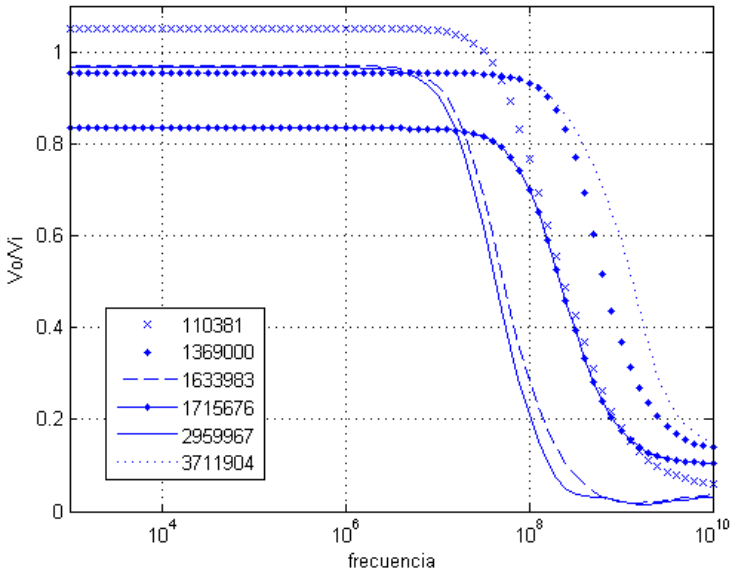

Figura 4.10. Ganancia de voltaje de los circuitos que se comportaron como VFs de 4 MOSFETs. Mostrados en la figura 4.9
(a) 110381, (b) 1369000, (c) 1633983, (d) 1715676, (e) 2959967 y (f) 3711904.

4.3 Topologías de CFs generadas por el GA.

Para ejecutar el programa que genera topologías de CFs se teclea en MatLab ‘sintesisCF’; y este comienza generando una población inicial de 4 individuos, indistintamente si se trata de un CF de 1, 2 ó 4 MOSFETs. El tamaño de la población siempre se conserva; pero si se modifica el programa para comenzar con una población inicial mayor a 4, las siguientes generaciones pueden decrecer asta un mínimo de 4 individuos. En la figura 4.11 se muestra el tiempo de ejecución (en segundos); al ejecutar el programa que encuentra CFs de 4 MOSFETs. En la figura 4.12 se observa el resultado de la evaluación del mejor de los individuos de cada generación; donde ‘aptitud 1’ es el circuito con fuentes de corriente ideales y la ‘aptitud 2’ es el circuito con espejos de corriente. Cabe mencionar que el programa es detenido cuando algún individuo rebasa la ‘aptitud 2’ pedida, en este ejemplo Io= 0.7*Ii.

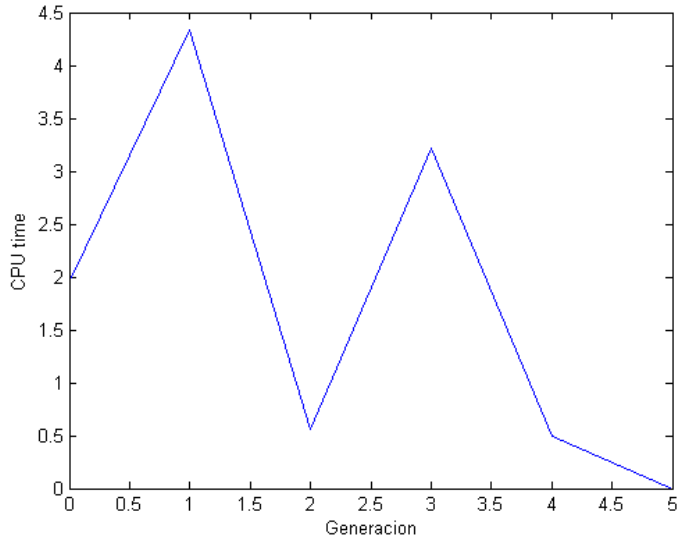

Figura 4.11. Tiempo de ejecución del programa CF de 4 MOSFETs.

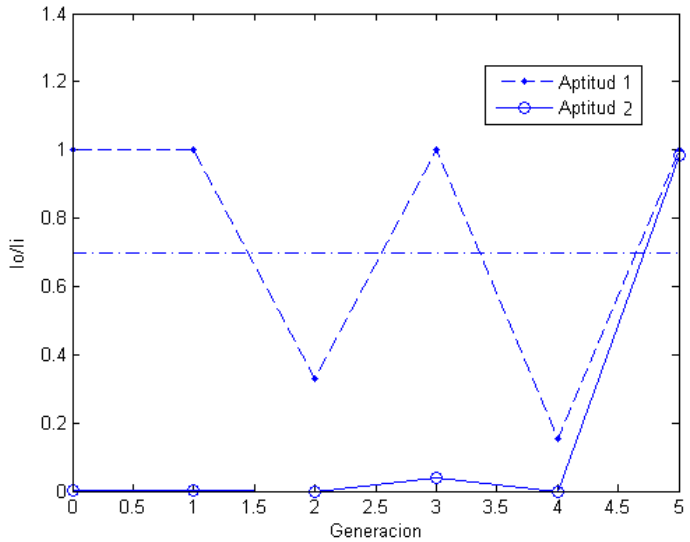

Figura 4.12. Medida de aptitud de los individuos en diferentes generaciones.

En la figura 4.13 se muestran 4 circuitos que se comportaron como CFs de 1 MOSFET (a partir de la figura 3.3(a)). En la figura 4.14 se observan 4 circuitos que se comportaron como CFs de 2 MOSFETs (a partir de la figura 3.3(b)); y en la figura 4.15 se exponen 6 circuitos que se comportaron como buenos CFs de 4 MOSFETs (a partir de la figura 3.3(c)). En la figura 4.16 se muestra la ganancia de corriente de los circuitos de la figura 4.15 (a) 1472026, (b) 1663488, (c) 2856163, (d) 2872546, (e) 3312240 y (f) 3830426.

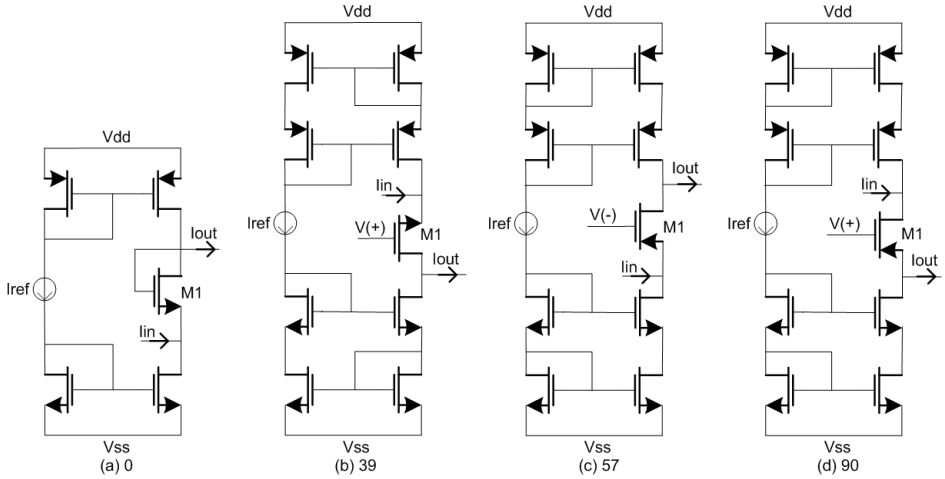

Figura 4.13. Circuitos que se comportaron como CFs de 1 MOSFET.

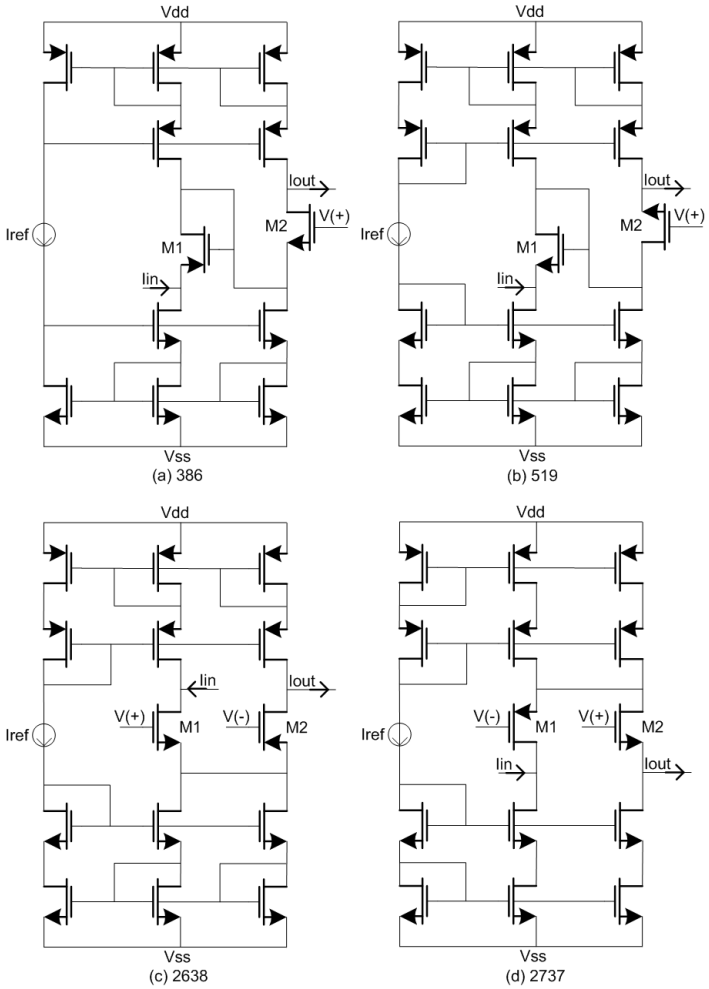

Figura 4.14. Circuitos que se comportaron como CFs de 2 MOSFETs.

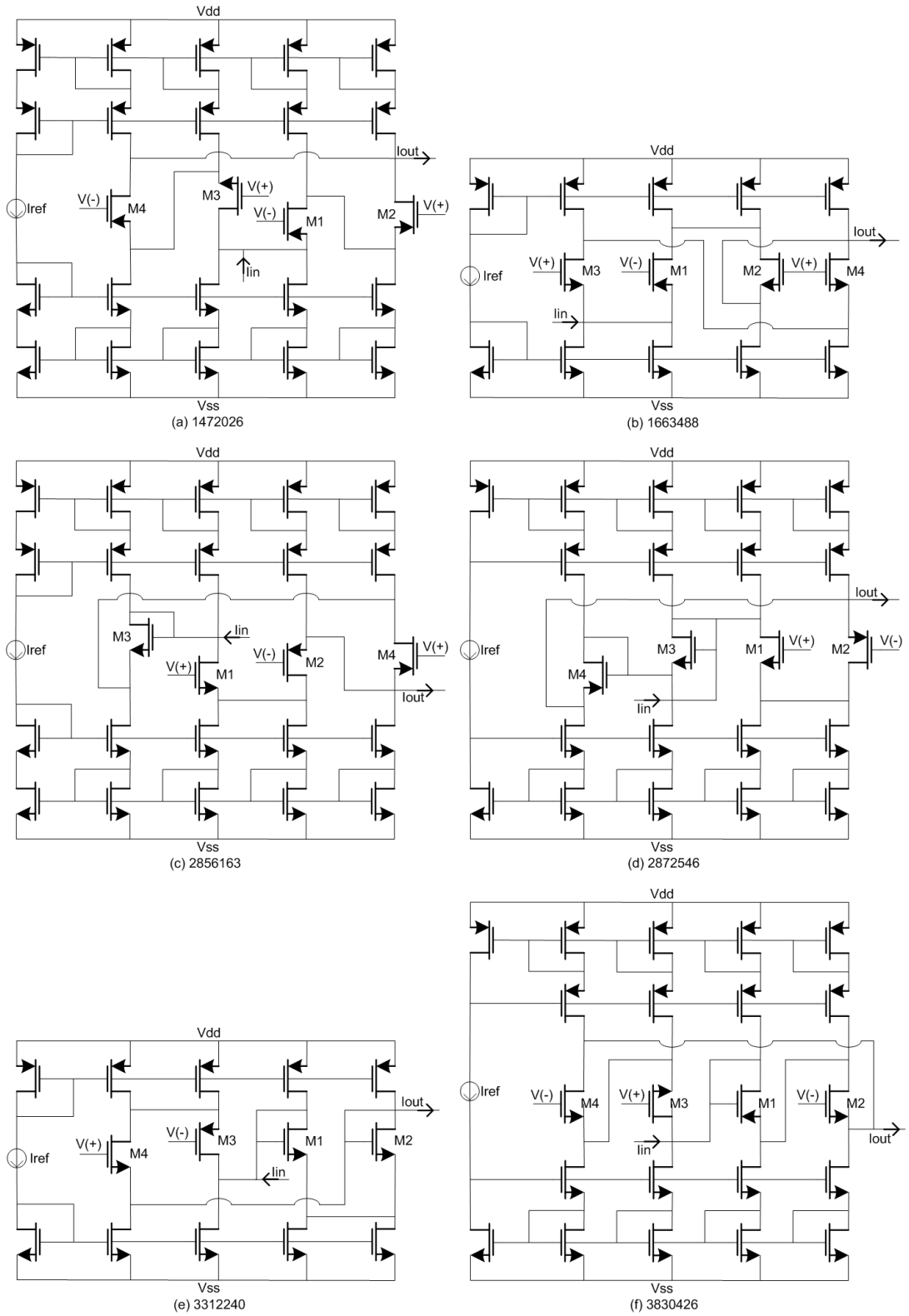

Figura 4.15. Circuitos que se comportaron como CFs de 4 MOSFETs.

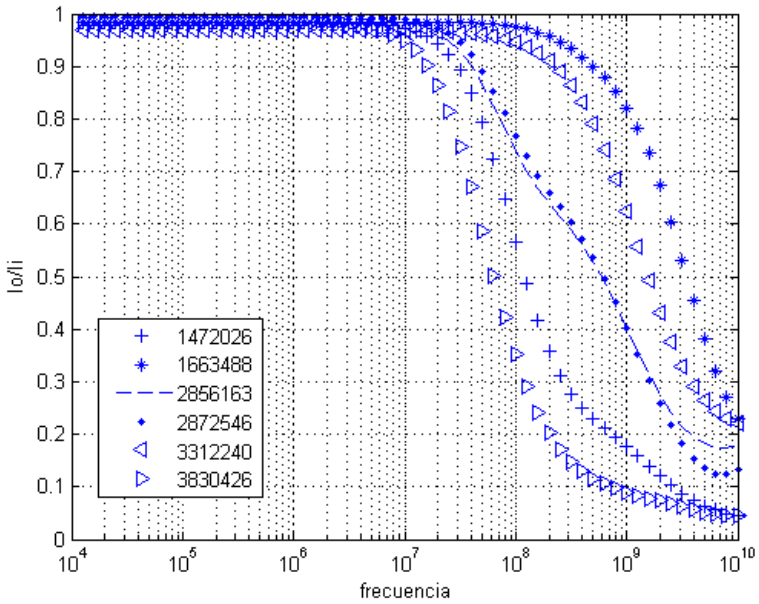

Figura 4.16. Ganancia de corriente de los circuitos que se comportaron como CFs de 4 MOSFETs, mostrados en la figura 4.15.
(a) 1472026, (b) 1663488, (c) 2856163, (d) 2872546, (e) 3312240 y (f) 3830426.

4.4 Topologías de VMs generadas por el GA.

Al teclear en MatLab la instrucción ‘sintesisVM’ se ejecuta el programa que encuentra topologías de VMs; este guarda mucha similitud con el programa que encuentra topologías de VFs. El tiempo de ejecución es similar a los programas anteriormente mostrados, aproximadamente en un minuto se encuentra una topología.

En la figura 4.17 se muestran 4 circuitos que se comportaron como VMs de 1 MOSFET (a partir de las figuras 3.2(a) y 3.16). En la figura 4.18 se observan 4 circuitos que se comportaron como VMs de 2 MOSFETs (a partir de las figuras 3.2(b) y 3.16). En la figura 4.19 se presentan 6 circuitos que se comportaron como VMs de 4 MOSFETs (a partir de las figuras 3.2(c) y 3.16), (a) 220762, (b) 3267966, (c) 6551537, (d) 7146019, (e) 8220493 y (f) 8217159. En la figura 4.20 se muestra la ganancia de voltaje de los circuitos de la figura 4.19.

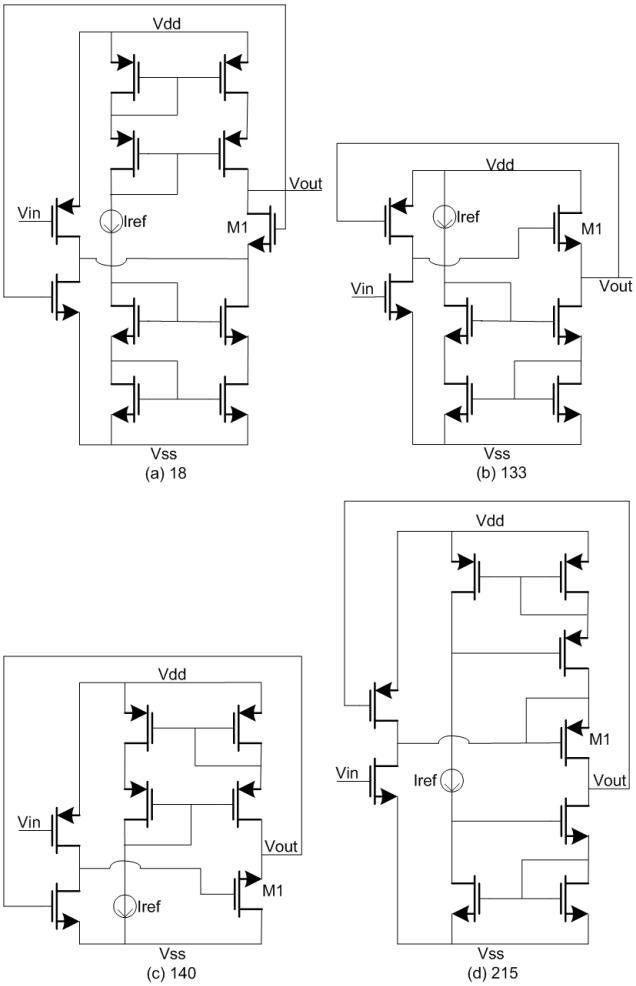

Figura 4.17. Circuitos que se comportaron como VMs de 1 MOSFET.

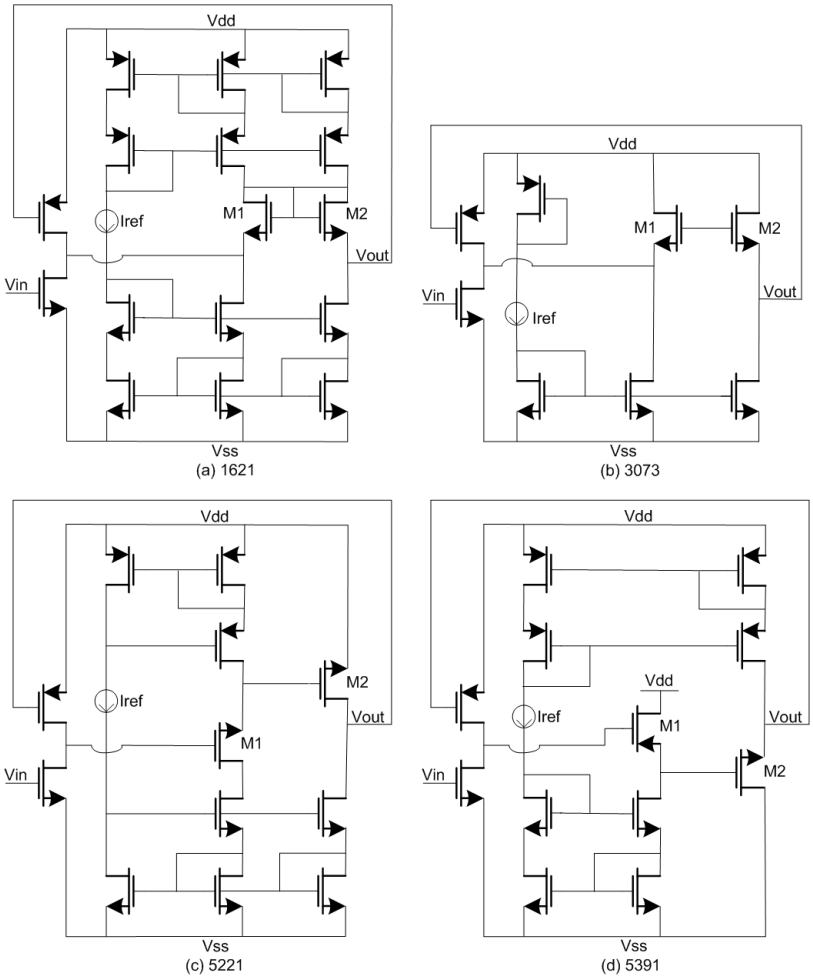

Figura 4.18. Circuitos que se comportaron como VMs de 2 MOSFETs.

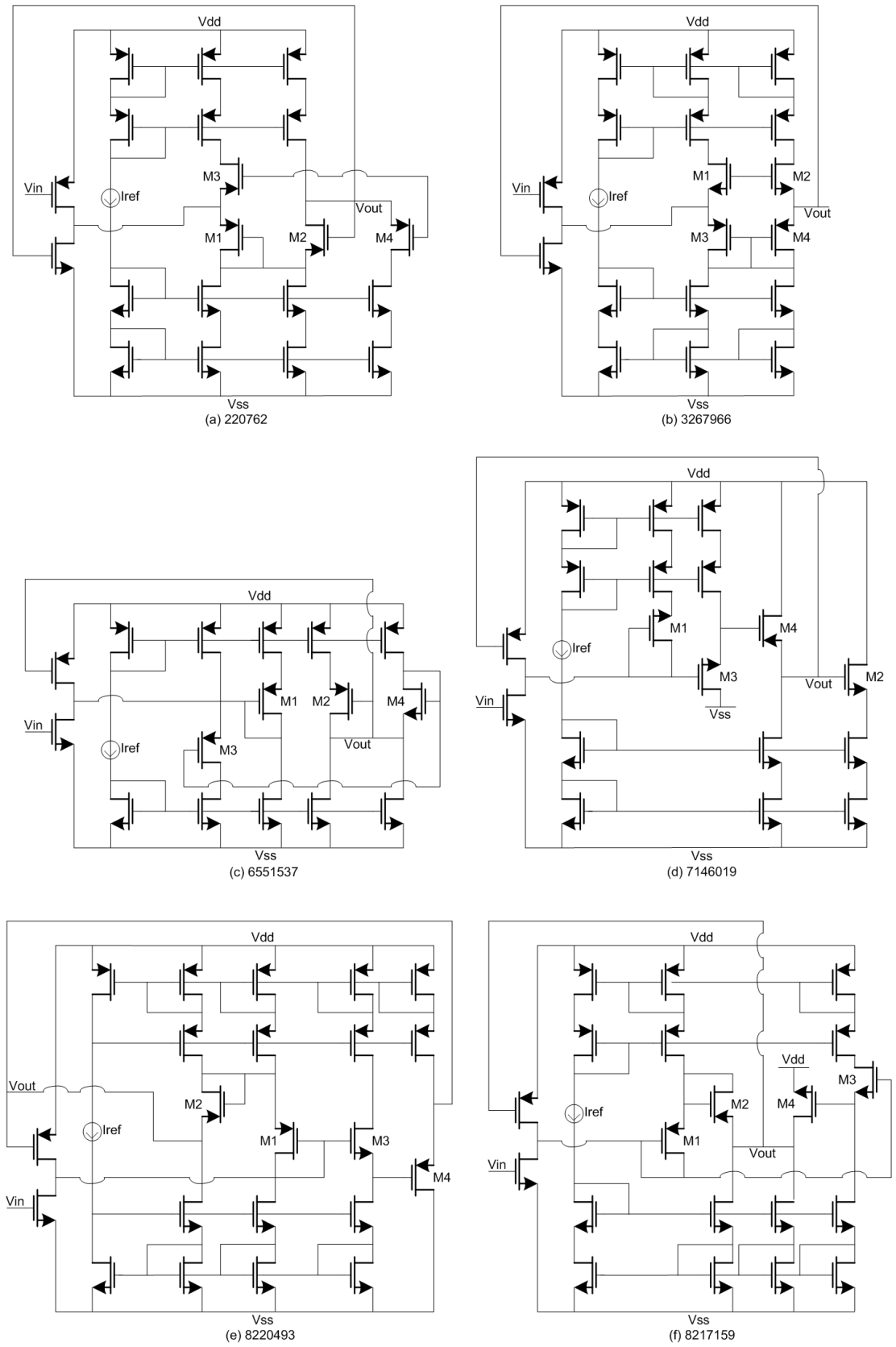

Figura 4.19. Circuitos que se comportaron como VMs de 4 MOSFETs.

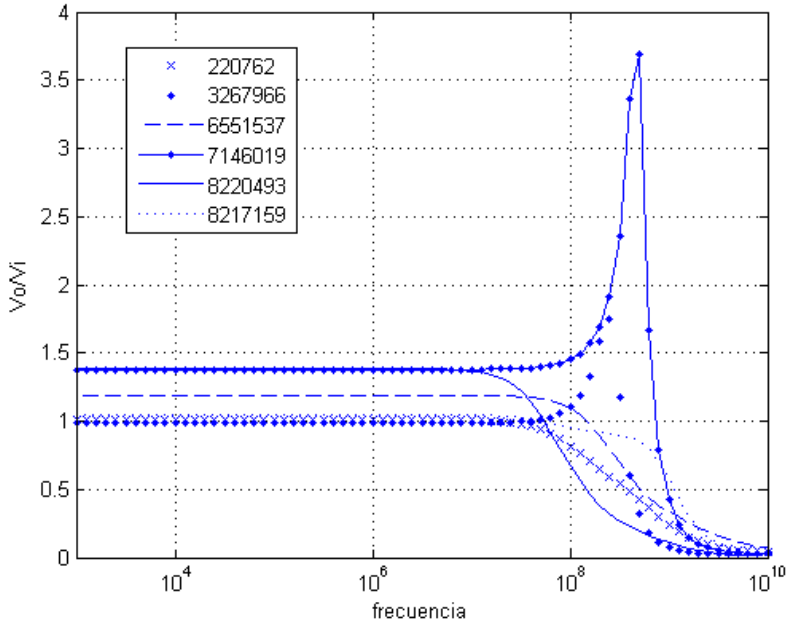

Figura 4.20. Ganancia de voltaje de los circuitos que se comportaron como VMs de 4 MOSFETs, mostrados en la figura 4.19
(a) 220762, (b) 3267966, (c) 6551537, (d) 7146019, (e) 8220493 y (f) 8217159.

4.5 Topologías de CMs generadas por el GA.

Al teclear en MatLab la instrucción ‘sintesisCM’ se ejecuta el programa que encuentra circuitos espejos de corriente. El tiempo de ejecución es similar a los programas anteriores.

En la figura 4.21 se muestran los únicos 4 circuitos que se comportaron como buenos CMs de 2 MOSFETs (a partir de la figura 3.19(a)); como se observa en la figura 4.21 un circuito puede ser representado por más de un código genético, esto se debe a la forma en como se realizo la decodificación genética. En las figuras 4.22 y 4.23 se observan 8 circuitos que se comportaron como CMs de 3 MOSFETs, a partir de las figuras 3.19(b) y 3.19(c), respectivamente. En la figura 4.24 se presentan 4 circuitos que se comportaron como CMs de 4 MOSFETs (a partir de la figura 3.19(d)). En la figura 4.25 se muestra la ganancia de corriente de los circuitos de la figura 4.24 (a) 282, (b) 6979, (c) 7710 y (d) 14743. En esta figura 4.25 se observa claramente que el simulador SPICE realiza una respuesta errónea de los circuitos 282, 7710 y 14743; así para el programa en MatLab estos están funcionando como CF. Mientras que un

diseñador se puede dar cuenta del error, éste es un ejemplo claro de que es necesario incluir reglas basadas en la experiencia de los diseñadores. De esta manera, en un trabajo a futuro debe MatLab no solo seleccionar el circuito por su respuesta en ganancia en CA sino por un conjunto de parámetros en CD, CA y su respuesta en el tiempo.

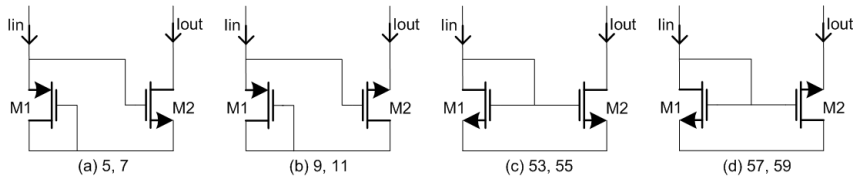

Figura 4.21. Circuitos que se comportaron como CMs de 2 MOSFETs.

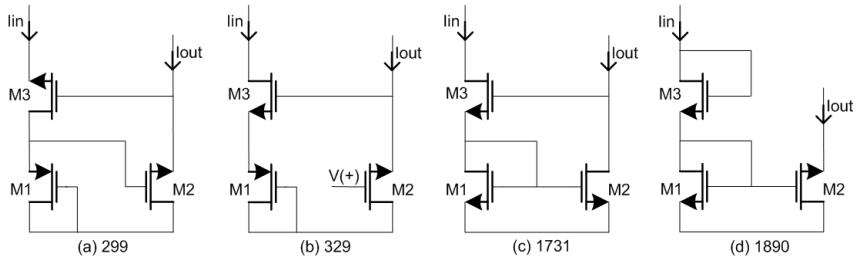

Figura 4.22. Circuitos que se comportaron como CMs de 3 MOSFETs a partir de la figura 3.19(b).

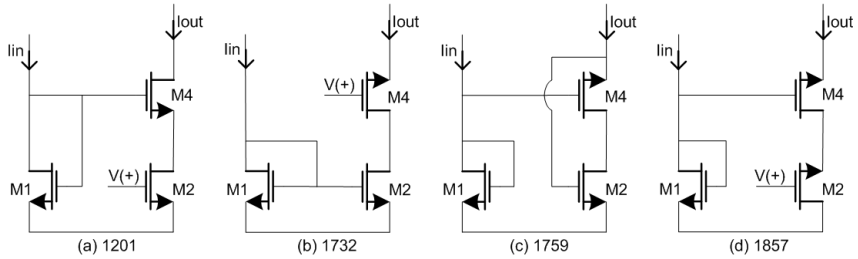

Figura 4.23. Circuitos que se comportaron como CMs de 3 MOSFETs a partir de la figura 3.19(c).

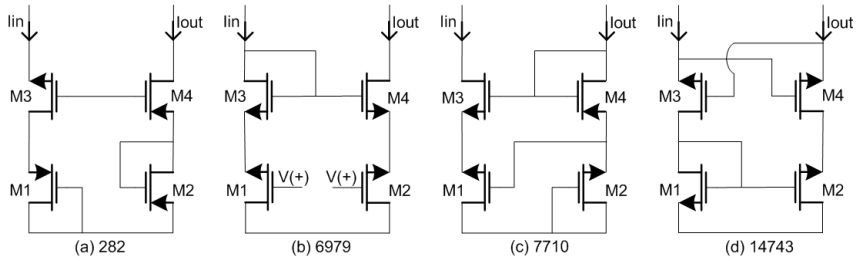

Figura 4.24. Circuitos que se comportaron como CMs de 4 MOSFETs.

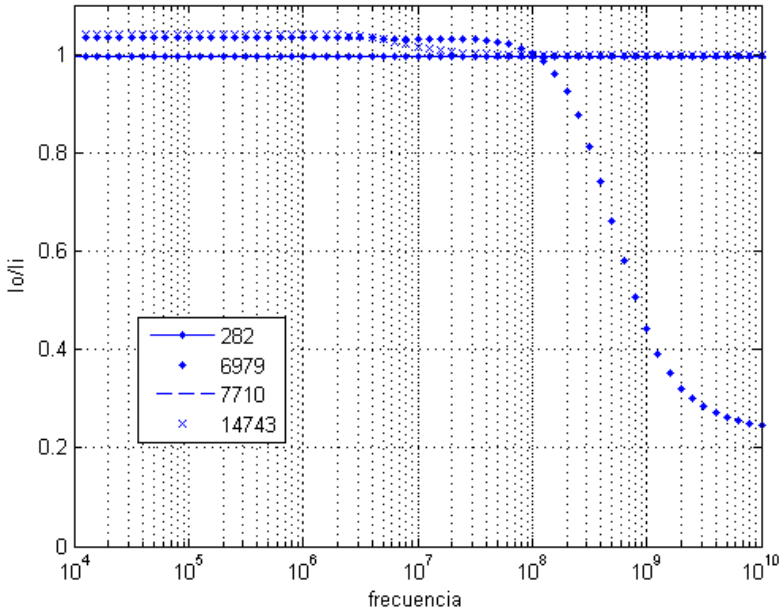

Figura 4.25. Ganancia de corriente de los circuitos que se comportaron como CMs de 4 MOSFETs, mostrados en la figura 4.24.
(a) 282, (b) 6979, (c) 7710 y (d) 14743.

4.6 Evolución de los cromosomas.

En esta sección se muestra como puede extenderse el código genético realizado para los bloques de ganancia unitaria y representar circuitos más complejos como CC. Como se muestra en la figura 4.26(a), un CCII+ puede ser representado como un VF seguido de un CM [16,29,30]. Similarmente, como se observa en 4.26(b) al unir un VF con un CF se obtiene un CCII- [16,29,30,39,40]. Si ahora el VF es cambiado por un VM se obtienen los CC inversos como se muestra en las figuras 4.26(c) y 4.26(d).

De la figura 4.26 se deduce fácilmente que para realizar el cromosoma de un CC basta con unir dos cromosomas de los bloques de ganancia unitaria. Así:

Cromosoma CCII+ => Cromosoma VF – Cromosoma CM

Cromosoma CCII– => Cromosoma VF – Cromosoma CF

Cromosoma ICCII+ => Cromosoma VM – Cromosoma CM

Cromosoma ICCII– => Cromosoma VM – Cromosoma CF

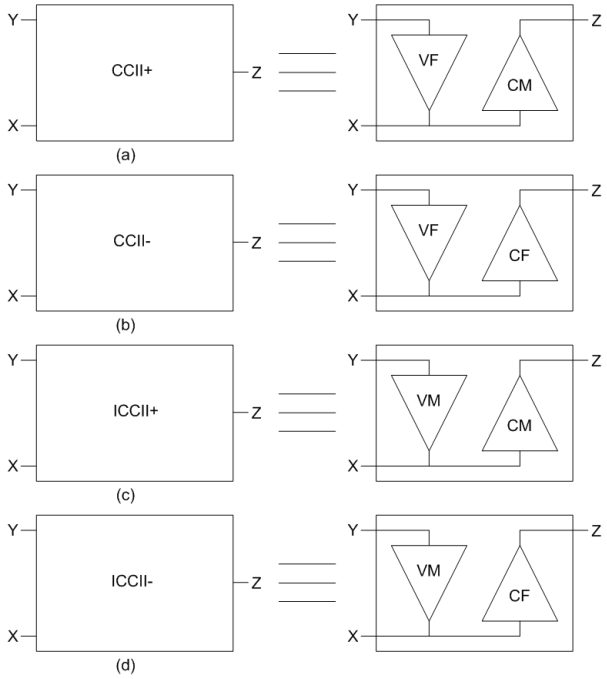

Figura 4.26. Arquitectura de (a) CCII+, (b) CCII-, (c) ICCII+ y (d) ICCII-.

Sin embargo, si se realiza una súper-imposición de un VF y un CF [2,36,41] se obtiene un CCII-, como se observa en la figura 4.27. Para representar este circuito con un código genético basta con tomar el código realizado para un VF y agregarle un genSI que represente la súper-imposición. Este genSI representaría una biblioteca de CFs que se pueden súper-imponer al circuito VF realizado.

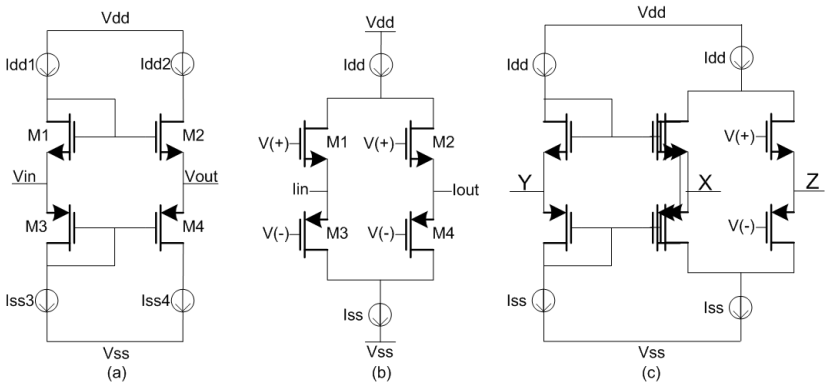

Figura 4.27. Súper-imposición de (a) figura 3.10 y (b) figura 3.14; para generar un CCII-.

Similarmente, al agregar un gen de súper-imposición (genSI) al código de un VM se obtendría la representación genética de un ICCII-. Realizando el incrmento del genSI los cromosomas quedan de la siguiente manera:

Cromosoma CCII- => genSS genSMos genBias genCM genSI,

Cromosoma ICCII- => genSS genSMos genBias genCM genInv genSI, donde el genSI representaría una biblioteca de CFs y los otros genes ya han sido explicados en el capítulo 3.

En la figura 4.28(a) se muestra a un circuito VF conocido (figura 3.10), si las fuentes de polarización Idd2 e Idd4 se sintetizan como se muestra en la figura 4.28(b) se obtiene el nodo Z para realizar la configuración de un CCII+. Entonces, para realizar la codificación genética de un CCII+ basta con agregar un genCM2 al código de un VF. Este genCM2 sintetiza las fuentes de polarización de M2 y M4, de una biblioteca de espejos de corriente; mientras el genCM solo sintetizaría las fuentes de polarización de los transistores M1 y M3.

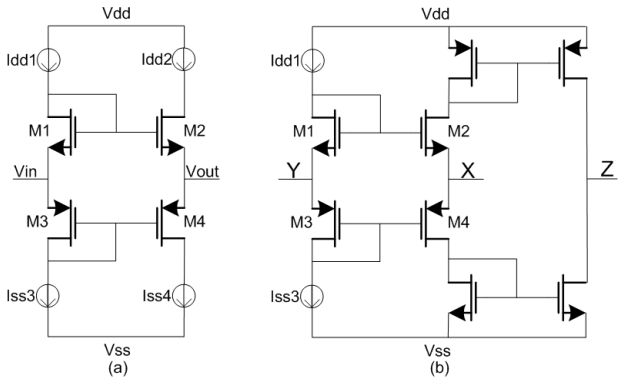

Figura 4.28. (a) VF conocido figura 3.10, (b) CCII+.

De igual forma al agregar un genCM2 al cromosoma de un VM se obtendría la representación genética de un ICCII+. Agregando el genCM2 a los códigos de los VFs y VMs se tiene:

Cromosoma CCII+ => genSS genSMos genBias genCM genCM2,

Cromosoma ICCII+ => genSS genSMos genBias genCM genInv genCM2, donde el genCM2 representaría una biblioteca de CMs para sintetizar el nodo Z del CC.

Por otra parte, un amplificador operacional retroalimentado en corriente (CFOA) puede ser representado por un CCII+ en serie con un VF, como se muestra en la figura 4.29 [29,30,36,40]. De la figura 4.29 se deduce fácilmente que para realizar el cromosoma de un CFOA basta con unir tres cromosomas de los bloques de ganancia unitaria dos VFs y un CF. Sin embargo, si se realiza una súper-imposición de un VF y un CF, se tiene el cromosoma de un CCII+; y si a este cromosoma se le agrega la codificación de un segundo VF se obtiene el código genético de un CFOA; como se observa en el diagrama de la figura 4.30.

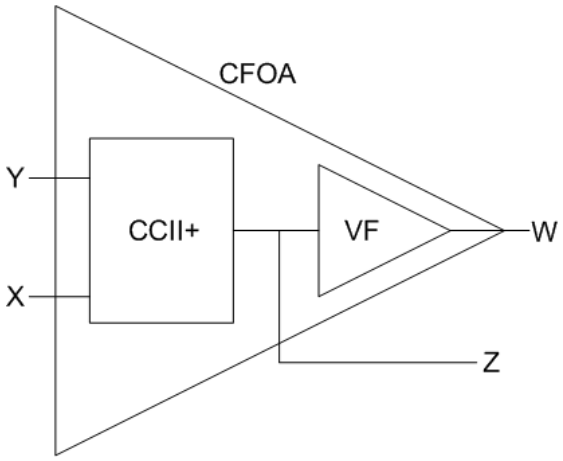

Figura 4.29. Diagrama a bloques de un CFOA.

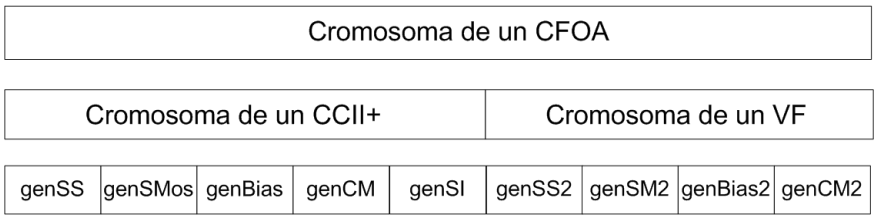

Figura 4.30. Diagrama a bloques de los genes de un cromosoma de un CFOA.

Capítulo 5

Conclusiones.

5.1 Conclusiones.

Se presentó la propuesta de un sistema automático para sintetizar bloques de ganancia unitaria: VFs, VMs, CFs y CMs; utilizando algoritmos genéticos en MatLab y T-SPICE para evaluar su aptitud.

Se propuso un nuevo método de codificación binaria llamado cromosoma, para representar un bloque de ganancia unitaria. Este cromosoma es la unión de genes que modifican su tamaño según el número de elementos nullor usados para modelar el comportamiento ideal del bloque de ganancia unitaria.

Se presentó la base para realizar automáticamente el netlist de bloques de ganancia unitaria y su simulación en SPICE. Se describió la manera en que se discriminan las topologías que no se comportan como buenos bloques de ganancia unitaria, esto se realiza a través de tres evaluaciones: la primera para verificar si la topología es sintetizable, la segunda evalúa el circuito sintetizado con polarización ideal y la tercera valora el circuito completamente sintetizado con transistores MOS. Las topologías sintetizadas se diseñan con tecnología de CIs CMOS de 0.35µm.

El método propuesto de síntesis ha sido programado en MatLab y se mostraron algunas topologías sintetizadas que cumplieron con las especificaciones pedidas, aunque algunas no son practicas como se discutió en el capítulo 4. Se ejecutó el programa de síntesis automática y

se mostraron algunas de las topologías que cumplieron con la especificación Vo>A*Vi ó Io>A*Ii según el caso voltaje o corriente.
Además, se propone como evolucionar el cromosoma para sintetizar otros circuitos más complejos como CCs y CFOAs.

5.2 Trabajo a futuro.

Un trabajo futuro claramente es el de optimizar los circuitos encontrados para lograr que Vo=Vi, Io=Ii, Vo=-Vi, Io=-Ii; según cada circuito VF, CF, VM y CM.
Los parámetros de evaluación para la selección de los circuitos (medida de aptitud) pueden crecer, según alguna especificación en CD, CA y/o análisis en el tiempo y no solo limitarse a ganancia en CA, única utilizada en esta tesis.
Extender el algoritmo genético para sintetizar CCs por manipulación de los bloques de ganancia unitaria.
Extender el algoritmo genético para sintetizar CFOAs por interconexión de CCs y VFs.

Referencias

[1] Rutenbar Rob A, Georges G. E. Gielen, and Brian A. Antao. Computer-Aided Design of Analog Integrated Circuits and Systems. IEEE Press. USA, 2002.

[2] Torres-Papaqui L., Torres-Muñoz D. y Tlelo-Cuautle E. Synthesis of VFs and CFs by Manipulation of Genetic Cells. Analog Integrated Circuits and Signal Processing. Vol.46, no.2, pp. 99-102, 2006.

[3] Mattiussi, Claudio. Evolutionary synthesis of analog networks. Doctoral thesis. Laboratory of Intelligent Systems, Institute of System Engineering Ecole Polytechnique Federale de Lausanne (EPFL), 2005.

[4] Mazumder, Pinaki; Elizabeth M. Rudnick. Genetic Algorithms for VLSI Design, Layout & Test Automation. Ed. Prentice Hall PTR. USA, 1999.

[5] Salem Zebulum, Ricardo; Marco Aurelio C. Pacheco, y Marley Maria B. R. Vellasco. Evolutionary Electronics, Automatic Design of Electronic Circuits and Systems by Genetic Algorithms. Ed. CRC Press. USA, 2002.

[6] Masanori Natsui, Yoshiaki Tadokoro, Naofumi Homma, Takafumi Aoki, and Tatsuo Higuchi. Synthesis of current mirrors based on evolutionary graph generation with transmigration capability. IEICE Electronics Express. Vol.4, No.3, pp. 88-93.

[7] Tathagato Rai Dastidar, P. P. Chakrabarti, and Partha Ray. A synthesis system for analog circuits based on evolutionary search and topological reuse. IEE Transactions on Evolutionary Computation. Vol.9, No.2, pp.211-224. April 2005.

[8] Varum Aggarwal. Evolving Sinusoidal Oscillators Using Genetic Algorithms. Proc. The 2003 NASA/DoD Conference on Evolvable Hardware. Chicago, USA, 2003, pp. 67-76.

[9] Somaya Kayed, Ramy Iskander, Sami Moussa, and Hani F. Ragai. Genetic algorithms based technology migration of 2nd generation current conveyors.

[10] Feng Wang,Yuanxiang Li, and Kangshun Li. Automated analog circuit design using two-layer genetic programming. In Applied Mathematics and Computation. Ed. ELSEVIER. USA, August 2006. pp.11.

[11] Grimbleby James B., Automatic analogue circuit synthesis using genetic algorithms. IEEE Proc. Circuits Devices Syst., Vol.147, No. 6: 319-323, Dec. 2000.

[12] Koza John R, Jones Lee W, Keane Martin A, Streeter Matthew J, Al-Sakran Sameer H. Toward automated design of industrial-strength analog circuits by means of genetic programming. In Genetic Programming Theory and Practice II. Boston: Kluwer Academic Publishers Chapter 8, pp. 121–142, 2004.

[13] Kumar P., and R. Senani. Bibliography on Nullors and Their Applications in Circuit Analysis, Synthesis and Design. Analog Integrated Circuits and Signal Processing, Vol. 33, pp. 65-76, 2002.

[14] Lawrence Davis. Handbook of genetic algorithms. Ed. Van Nostrand Reinhold. New York, 1991.

[15] Tlelo-Cuautle E., Torres-Muñoz D., and Torres-Papaqui L. On the computational synthesis of CMOS voltage followers. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences. Vol.E88-A, No.12, pp. 3479-3484, 2005.

[16] I. A. Awad, and A. M. Soliman. On the Voltage Mirrors and the Current Mirrors. Analog Integrated Circuits and Signal Processing. Vol.32, pp. 79-81. 2002.

[17] Hung-Yu Wang, Sheng-Hsiung Chang, Yuan-Long Jeang, Chun-Yueh Huang. Rearrangement of mirror elements. Analog Integr Circ Sig Process Vol 49, 2006. pp. 87-90.

[18] Samir Ben Salem, Mourad Fakhfakh, Dorra Sellami Masmoudi, Mourad Loulou, Patrick Loumeau, Nouri Masmoudi. A high performances CMOS CCII and high frecuency applications. Analog Integr Circ Sing Procees. Springer Science Business Media, LLC 2006.

[19] Shoou-Jinn Chang, Hao-Sheng Hou, and Yan-Kuin Su. Automated Passive Filter Synthesis Using a Novel Tree Representation and Genetic Programming. IEEE transactions on Evolutionary Computation. Vol.10, No.1, pp. 93-100. February 2006.

[20] Kumar P., and Senani R. A systematic realization of current mode universal biquad filters. International Journal of Electronics, Vol.93, No.9, September 2006. pp. 623-636.

[21] Gupta S.S., and Senani R. New voltage-model/current-mode universal biquad filter using unity-gain cells. International Journal of Electronics, Vol.93, No.11, November 2006. pp. 760-775.

[22] Aggarwal Varun. Novel Canonic Current Mode DDCC Based SRCO Synthesized Using a Genetic Algorithm. Analog Integrated Circuits Signal Processing, No.40, 2004. pp. 83-85.

[23] Senani R., and Gupta S.S. Novel sinusoidal oscillators using only unity-gain voltage followers and current followers. IEICE Electronics Express, Vol.1, No.13, pp. 404-409. 2004.

[24] Gupta S.S., and Senani R. New single resistance controlled oscillator employing a reduced number of unity-gain cells. IEICE Electronics Express, Vol.1, No.16, pp. 507-512. 2004.

[25] Tlelo-Cuautle Esteban, Duarte-Villaseñor Miguel A, García-Ortega Johana M, Sánchez-López Carlos. Designing SRCOs by combining SPICE and Verilog-A. International Journal of Electronics, Vol.94, No.4, pp. 373-379. 2007.

[26] Tlelo-Cuautle E, Gaona-Hernández A, García-Delgado J. Implementation of a chaotic oscillator by designing Chua's diode with CMOS CFOAs. Analog Integr Circ Sig Process, Vol.48, pp. 159-162. 2006.

[27] Stefanovic Danica, Kayal Maher, and Pastre Marc. PAD: A New Interactive Knowledge-Based Analog Design Approach. Analog Integrated Circuits and Signal Processing, Vol.42, pp. 291-299, 2005.

[28] Chris Toumazou, George Moschytz, Barrie Gilbert, Trede-offs in Analog Circuit Design, Kluwer Academic Publishers, 2002

[29] Palmesano, Giuseppe; Gaetano Palumbo, and Salvatore Pennisi. CMOS Current Amplifiers. Kluwer Academic Publishers. USA, 1999.

[30] Sedra A. S., Roberts G. W., and Gohn F. The current conveyor: history, progress and new result. IEE proceedings, Vol.137. No.2. pp. 78-86. April 1990.

[31] Alfonso L. de Garay. Genética de poblaciones y evolución. Comportamiento de los genes en las poblaciones y efectos de las fuerzas de la evolución biológica. Ed. textos-uap. México, 1988.

[32] Vázquez y Montiel, Sergio. Diseño de sistemas ópticos usando algoritmos genéticos. Tesis doctoral. Instituto Nacional de Astrofísica, Óptica y Electrónica. Tonanzintla, Puebla, México, 1996.

[33] Salazar Romero, Marcos Arturo. Obtención de la fase a partir de la función de punto extendido utilizando un algoritmo genético. Tesis doctoral. Instituto Nacional de Astrofísica, Óptica y Electrónica. Tonanzintla, Puebla, México, 2005.

[34] Santos Gordillo, José Ángel. Algoritmo genético para el cálculo de Φ-Tensores difusos. Tesis de Maestría en Ciencias. Instituto Nacional de Astrofísica, Óptica y Electrónica. Tonanzintla, Puebla, México, 2003.

[35] Hanspeter Schmid. Approximating the Universal Active Element. IEEE Trans. On Circuits and Systems II. Vol.47, No.11, pp.1160-1169, 2000.

[36] Tlelo-Cuautle E., Torres-Muñoz D., Torres-Papaqui L., Muñoz-Pacheco M, Gaona-Hernández A., and García-Delgado J. Synthesis of CCII+s and CFOAS by manipulation of VFs and CMs. IEEE BMAS, web-publication, sep 2005. www.bmas-conf.org/2005/

[37] Sedra A. y K Smith. Circuitos Micro-electrónicos. 4ta. Edición, Ed. Oxford. México, 1999.

[38] Muhammed A. Ibrahim, and Hakan Kuntman. A CMOS realization of Inverting second generation current conveyor positive. 5th Nordic Signal Processing Symposium, October 4-7 2002.
www.norsig.no/norsig2002/Proceedings/papers/cr1055.pdf

[39] Torres Papaqui, Leticia. Síntesis de Dispositivos Activos Vía Manipulación de Celdas Genéricas. Tesis de Maestría en Ciencias. Instituto Nacional de Astrofísica Óptica y Electrónica. Tonanzintla, Puebla, México, 2004.

[40] Peña Pérez Aldo. Diseño de CFOAS en tecnología CMOS. Tesis de Maestría en Ciencias. Instituto Nacional de Astrofísica Óptica y Electrónica. Tonanzintla, Puebla, México, 2006.

[41] Hajime Shibata, and Nobuo Fujii. Analog circuit synthesis by superimposing of sub-circuits. IEEE ISCAS, Vol. V, pp. 427-430, 2001.

Lista de figuras.

Lista de tablas.

Biografía

Miguel Aurelio Duarte Villaseñor

Información personal y profesional.

Yo nací el 14 de enero de 1981, en Puebla de los Ángeles, México. Me gradué de licenciatura en electrónica en la Universidad Autónoma de Puebla (BUAP), con promedio de 9.1. Mi especialidad de estudios fue en diseño de sistemas digitales; el trabajo de tesis presentado fue: 'Sistema de medición automático para la caracterización de microestructuras', en esta tesis se desarrollaron dos trabajos principales. El primero presenta un sistema automático para la caracterización de la impedancia de la interfaz microelectrodo/electrolito; esta interfaz esta presente en los microelectrodos utilizados en la medición de la corriente iónica en neuronas. El sistema es de propósito general; es decir, con la inversión de poco tiempo en la programación es viable adaptar subprogramas para la caracterización de otro tipo de circuitos. Este trabajo se presentó en la 2da Convención Nacional de Investigación Aplicada y Desarrollo Tecnológico 2006 en Puebla, obteniendo el 3er lugar Nacional en el área de Electrónica. El segundo trabajo desarrollado en la tesis de licenciatura fue un sistema de medición automático para caracterización de la frecuencia de resonancia de trampolines cerámicos. Este permite la obtención de la frecuencia de resonancia, extraída de la impedancia manifiesta en un circuito divisor de voltaje. Los resultados se han utilizado para determinar el peso de un material (cientos de nanogramos) a través de una variación en la frecuencia de resonancia. Este sistema obtuvo el 3er lugar Nacional en el área de Mecatrónica en la ya mencionada convención. Actualmente, realizo los estudios de Maestría en Ciencias de la Electrónica en el Instituto Nacional de Astrofísica Óptica y Electrónica (INAOE), culminando con la presente tesis.

Apéndice A

Publicaciones.

Las publicaciones marcadas con el símbolo & han sido desarrolladas sobre este trabajo de tesis.

A.1 Publicaciones en revistas.

E. Tlelo-Cuautle, M. A. Duarte-Villaseñor, and J. M. García-Ortega. Modelado y Simulación de un Oscilador Caótico usando MatLab. IEEE Latin America Transactions, Vol.5, No.2, May 2007, pp. 95-98.

E. Tlelo-Cuautle, M. A. Duarte-Villaseñor, J. M. García-Ortega, C. Sánchez-López, Designing SRCOs by combining SPICE and Verilog-A, International Journal of Electronics, Vol.94, No.4, April 2007, pp. 373-379, ISSN: 0020-7217.

E. Tlelo-Cuautle, M. A. Duarte-Villaseñor. 'Designing Chua's circuit from the behavioral to the transistor level of abstraction'. Applied Mathematics and Computation vol.184. Ed. Elsevier. January 2007. pp. 715–720.

A.2 Publicaciones en congresos.

& M. A. Duarte-Villaseñor, E. Tlelo-Cuautle, G. Reyes-Salgado y C. A. Reyes-García. 'Síntesis Automática de Seguidores de Voltaje Usando Algoritmos Genéticos'. En el VII Congreso Nacional de Ingeniería Eléctrica y Electrónica del Mayab (CONIEEM 2007). Mérida, Yucatán, México. 26-30 de marzo de 2007.

Resumen: Se describe un método de síntesis automática de VFs basado en la aplicación de GAs. Los VFs son diseñados con tecnología de circuitos integrados CMOS de 0.35μm. Se muestra la utilidad del elemento anulador para modelar el comportamiento ideal del VF, y para codificar su topología utilizando un cromosoma dividido en cuatro genes: gen de pequeña señal (genSS), gen de síntesis del MOSFET (genSMos), gen de polarización (genBias) y gen de síntesis de espejos de corriente (genCM). El método de síntesis propuesto se programó en MatLab y usa T-SPICE para verificar el comportamiento de los VFs, los cuales se seleccionan por elitismo.

Duarte-Villaseñor Miguel Aurelio, García-Ortega Johana María, Soto-Cruz Blanca Susana y Alcántara-Iniesta Salvador. 'Sistema de medición automático para caracterización de la impedancia de la interfaz microelectrodo/electrolito'. En la 2da Convención Nacional de Investigación Aplicada y Desarrollo Tecnológico 2006. Puebla, Puebla, México, 16-17 de noviembre de 2006.

Este trabajo presenta un sistema automático para la caracterización de la impedancia de la interfaz microelectrodo/electrolito; esta interfaz esta presente en los microelectrodos utilizados en la medición de la corriente iónica en neuronas. El sistema es de propósito general; es decir, con la inversión de poco tiempo en la programación es viable adaptar subprogramas para la caracterización de otro tipo de circuitos. Cabe mencionar que esta presentación obtuvo el 3er lugar nacional en el área de Electrónica.

Duarte-Villaseñor Miguel Aurelio, García-Ortega Johana María, Soto-Cruz Blanca Susana y Alcántara-Iniesta Salvador. 'Sistema de medición automático para caracterización de la frecuencia de resonancia de trampolines cerámicos'. En la 2da Convención Nacional de Investigación Aplicada y Desarrollo Tecnológico 2006. Puebla, Puebla, México, 16-17 de noviembre de 2006.

Se realizo un sistema de caracterización sencillo, para encontrar la frecuencia de resonancia de trampolines cerámicos. Este sistema permite la obtención de la frecuencia de resonancia, extraída de la impedancia manifiesta en un circuito divisor de voltaje. Los resultados se han utilizado para determinar el peso de un material a través de una variación en la frecuencia de resonancia. Cabe mencionar que esta presentación consiguió el 3er lugar nacional en el área de Mecatrónica.

E. Tlelo-Cuautle, M. A. Duarte-Villaseñor, J. M. García-Ortega. 'Modeling and Simulation of a Chaotic Oscillator by MatLab'. In 'Electronics, Robotics, and Automotive Mechanics Conference, CERMA 2006, Volume I'. Cuernavaca, Morelos, México, September 26-29, 2006. pp. 239-242. ISBN 0-7695-2569-5.

Es presentado un sistema basado en MatLab para modelar y simular un oscilador caótico, el sistema se basa en resolver las ecuaciones de estado del circuito de Chua, graficando sus tres respuestas en transitorio y sus respuestas caóticas en 2D y 3D. Asimismo, este trabajo fue reconocido como uno de los diez mejores trabajos entre más de 180.

Alcántara Iniesta Salvador; Soto Cruz B. Susana, Duarte Villaseñor Miguel A. 'Sistema de medición automático para caracterización de microestructuras'. En el XV Congreso Interuniversitario de Electrónica, Computación y Eléctrica (CIECE2005). Puebla, Puebla, México. 7-9 de marzo de 2005.

García Ortega Johana María y Duarte Villaseñor Miguel Aurelio. Motor de Corriente Directa controlado por Voz. En el XV Congreso Interuniversitario de Electrónica, Computación y Eléctrica (CIECE2005). Puebla, Puebla, México. 7-9 de marzo de 2005.

Duarte Villaseñor Miguel Aurelio y García Ortega Johana María. 'Circuito contador tolerante a fallas'. En el XV Congreso Interuniversitario de Electrónica, Computación y Eléctrica (CIECE2005). Puebla, Puebla, México. 7-9 de marzo de 2005.

A.3 Publicaciones aceptadas.

& E. Tlelo-Cuautle, M.A. Duarte-Villaseñor, C.A. Reyes-García, C. Sánchez-López, M. Fakhfakh, M. Loulou, G. Reyes-Salgado. Designing VFs by applying genetic algorithms from nullator-based descriptions.

Congreso ECCTD, a realizarse en agosto de 2007, Sevilla, España.

& Esteban Tlelo-Cuautle and Miguel A. Duarte-Villaseñor. Evolutionary Electronics: Automatic Synthesis of Analog Circuits by GAs.

Capítulo del libro "Success in Evolutionary Computation".

& Esteban Tlelo-Cuautle, Miguel A. Duarte-Villaseñor, Carlos A. Reyes-García, Gerardo Reyes-Salgado. Automatic synthesis of electronics circuits using genetic algorithms.

Revista Computación y Sistemas, 2007.

Apéndice B

Códigos utilizados en SPICE.

A lo largo de esta tesis, diferentes códigos de cromosomas se han utilizado para describir circuitos. En este apéndice se muestra un netlist que fue realizado por el programa que genera VFs de 4 CMOS; este muestra en esencia como son los listados para SPICE realizados por los programas ejecutados. También se muestran los parámetros de la tecnología de circuitos integrados CMOS de 0.35µm, utilizados por las topologías sintetizadas.

B.1 Netlist utilizado para VFs de 4 MOSFETs.

Como se describió en el capítulo 4, el programa genera dos netlist para cada cromosoma que describe un circuito con la capacidad de supera la primera medida de aptitud. Como se menciona en la tesis, primero es generado un archivo para probar el circuito con fuentes ideales y después un netlist para el mismo circuito con espejos de corriente. Sin embargo, es creado también un listado llamado 'ArcFijo', el cual contiene información que no varia para todos los circuitos, así los programas realizados ahorran tiempo al generar los archivos para SPICE. En el listado B.1 se muestra la estructura del archivo fijo (ArcFijo); contiene las fuentes de alimentación del voltaje, la fuente de entrada, la carga utilizada (0.1pF). En sub-circuitos tanto en NMOS como en PMOS los espejos: simple, cascode, Wilson y Wilson modificado; también incluye la librería de parámetros de simulación y la instrucción para realizar la simulación en AC.

Listado B.1. Archivo fijo para los circuitos de ganancia de voltaje con 4 MOS.

```
VDD vdd GND 1.65
VSS GND vss 1.65
VIN IN GND AC 1
CL OUT GND 0.1p
.param LN=0.9E-6 WN=12E-6 LP=0.9E-6 WP=18E-6
.param FI=100u
.subckt NPSimple j m vdd vss
Ibias j m FI
Mb1 j j vdd vdd PMOS L=LP W=WP
Mb2 m m vss vss NMOS L=LN W=WN
.ends
.subckt NPCascode a b c d vdd vss
Ibias b c FI
Mb1 a a vdd vdd PMOS L=LP W=WP
```

```
Mb2 b b   a vdd PMOS L=LP W=WP
Mb3 c c   d vss NMOS L=LN W=WN
Mb4 d d vss vss NMOS L=LN W=WN
.ends
.subckt NPWilson a b c d vdd vss
Ibias b c FI
Mb1 b a vdd vdd PMOS L=LP W=WP
Mb2 c d vss vss NMOS L=LN W=WN
.ends
.subckt NPWilsonM a b c d vdd vss
Ibias b c FI
Mb1 1 a vdd vdd PMOS L=LP W=WP
Mb2 b b   1 vdd PMOS L=LP W=WP
Mb3 c c   2 vss NMOS L=LN W=WN
Mb4 2 d vss vss NMOS L=LN W=WN
.ends
.subckt SimpleP X j vdd
M1 X j vdd vdd PMOS L=LP W=WP
.ends
.subckt SimpleN X m vss
M1 X m vss vss NMOS L=LN W=WN
.ends
.subckt CascodeP X a b vdd
M1 1 a vdd vdd PMOS L=LP W=WP
M2 X b   1 vdd PMOS L=LP W=WP
.ends
.subckt CascodeN X c d vss
M1 X c   1 vss NMOS L=LN W=WN
M2 1 d vss vss NMOS L=LN W=WN
.ends
.subckt WilsonP X a b vdd
M1 a a vdd vdd PMOS L=LP W=WP
M2 X b   a vdd PMOS L=LP W=WP
.ends
.subckt WilsonN X c d vss
M1 X c   d vss NMOS L=LN W=WN
M2 d d vss vss NMOS L=LN W=WN
.ends
.include 'C:\MATLAB7\work\VF6\mos35t.lib'
.AC DEC 10 1e2 1e6
```

En el listado B.2, se muestra el archivo para SPICE del circuito cuyo código es ‘3711904’. La primera línea es el número del cromosoma, esta es descartada en la simulación de SPICE, enseguida se encuentran los 4 CMOS que definen el bloque VF, estos están conectados según el significado del cromosoma. Siguen las fuentes de polarización de corriente y voltaje, se incluye el archivo fijo y se pide la salida en formato de texto.

Listado B.2. Archivo para SPICE de un VF con polarización ideal.

```
3711904
M1 IN IN 1 vdd PMOS L=LP W=WP
M2 4 1 OUT vss NMOS L=LN W=WN
M3 2 2 IN vss NMOS L=LN W=WN
M4 6 2 OUT vss NMOS L=LN W=WN
IDD1 vdd 1 FI
```

```
IDD2 vdd 4 FI
IDD3 vdd 2 FI
VD4 6 GND 1.65
ISS1 IN vss FI
ISS2 OUT vss FI
ISS3 IN vss FI
ISS4 OUT vss FI
.include 'C:\MATLAB7\work\VF6\NewFolder1\ArcFijo.sp'
.print "salida1.txt" v(OUT)
.END
```

El último archivo realizado por los programas es el circuito polarizado con espejos de corriente. Continuando con el ejemplo, se ve en el listado B.3 el circuito VF ‘3711904’ de 4 CMOS polarizado con espejos de corriente; parecido con el listado B.2 solo que las fuentes de corriente son cambiadas por los bloques de espejos simples NMOS y PMOS. Si el código significa otro tipo de espejo para el circuito solo se seleccionan los bloques correspondientes del archivo fijo.

Listado B.3. Archivo para SPICE de un VF polarizado con espejos de corriente.

```
3711904
M1 IN IN 1 vdd PMOS L=LP W=WP
M2 4 1 OUT vss NMOS L=LN W=WN
M3 2 2 IN vss NMOS L=LN W=WN
M4 6 2 OUT vss NMOS L=LN W=WN
XP1 1 j vdd SimpleP
XP2 4 j vdd SimpleP
XP3 2 j vdd SimpleP
VD4 6 GND 1.65
XN1 IN m vss SimpleN
XN2 OUT m vss SimpleN
XN3 IN m vss SimpleN
XN4 OUT m vss SimpleN
XEsp j m vdd vss NPSimple
.include 'C:\MATLAB7\work\VF6\NewFolder1\ArcFijo.sp'
.print "salida2.txt" v(OUT)
.END
```

B.2 Parámetros de 0.35µm utilizados.

```
*T46W SPICE BSIM3 VERSION 3.1 PARAMETERS
*SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8
* DATE: Sep 14/04
* LOT: T46W                  WAF: 1102
* Temperature_parameters=Default
.MODEL NMOS NMOS (                                LEVEL   = 49
+VERSION = 3.1           TNOM    = 27             TOX     = 8E-9
+XJ      = 1E-7          NCH     = 2.2E17         VTH0    = 0.4074928
+K1      = 0.4728294     K2      = 3.621074E-3    K3      = 70.0489524
+K3B     = -10           W0      = 1.830495E-5    NLX     = 2.193565E-7
+DVT0W   = 0             DVT1W   = 0              DVT2W   = 0
+DVT0    = 0.802594      DVT1    = 0.4276766      DVT2    = -0.3
+U0      = 361.3464355   UA      = -9.67751E-10   UB      = 2.889157E-18
+UC      = 4.669186E-11  VSAT    = 1.898742E5     A0      = 1.3381235
+AGS     = 0.2592162     B0      = 2.220067E-6    B1      = 5E-6
```

```
+KETA    = -9.077245E-3   A1      = 0              A2      = 0.3487525
+RDSW    = 780.376869     PRWG    = 0.0713836      PRWB    = -7.21866E-3
+WR      = 1              WINT    = 1.389179E-7    LINT    = 1.24319E-9
+DWG     = -1.034792E-8   DWB     = 9.824117E-9    VOFF    = -0.0869274
+NFACTOR = 0.7366118      CIT     = 0              CDSC    = 2.4E-4
+CDSCD   = 0              CDSCB   = 0              ETA0    = 0.0346215
+ETAB    = -7.988336E-3   DSUB    = 0.3317286      PCLM    = 1.9546403
+PDIBLC1 = 4.161735E-3    PDIBLC2 = 1.19743E-5     PDIBLCB = 0.1
+DROUT   = 2.748338E-3    PSCBE1  = 7.42428E8      PSCBE2  = 1E-3
+PVAG    = 0              DELTA   = 0.01           RSH     = 3.9
+MOBMOD  = 1              PRT     = 0              UTE     = -1.5
+KT1     = -0.11          KT1L    = 0              KT2     = 0.022
+UA1     = 4.31E-9        UB1     = -7.61E-18      UC1     = -5.6E-11
+AT      = 3.3E4          WL      = 0              WLN     = 1
+WW      = 0              WWN     = 1              WWL     = 0
+LL      = 0              LLN     = 1              LW      = 0
+LWN     = 1              LWL     = 0              CAPMOD  = 2
+XPART   = 0.5            CGDO    = 2.58E-10       CGSO    = 2.58E-10
+CGBO    = 1E-12          CJ      = 1.012513E-3    PB      = 0.8
+MJ      = 0.3510986      CJSW    = 2.862666E-10   PBSW    = 0.8
+MJSW    = 0.1518459      CJSWG   = 1.82E-10       PBSWG   = 0.8
+MJSWG   = 0.1518459      CF      = 0              PVTH0   = -0.0100437
+PRDSW   = -73.5674578    PK2     = 3.087074E-3    WKETA   = 3.003636E-3
+LKETA   = 2.647195E-3     )
*
.MODEL PMOS PMOS (                                LEVEL   = 49
+VERSION = 3.1            TNOM    = 27             TOX     = 8E-9
+XJ      = 1E-7           NCH     = 8.52E16        VTH0    = -0.6831778
+K1      = 0.3939412      K2      = 0.0308482      K3      = 0
+K3B     = 15.35112       W0      = 1E-5           NLX     = 1E-9
+DVT0W   = 0              DVT1W   = 0              DVT2W   = 0
+DVT0    = 1.4617205      DVT1    = 0.3569339      DVT2    = -0.0368562
+U0      = 221.3795636    UA      = 1.573901E-9    UB      = 5E-18
+UC      = 8.567678E-11   VSAT    = 2E5            A0      = 1.999067
+AGS     = 0.389467       B0      = 2.419633E-6    B1      = 5E-6
+KETA    = -6.020293E-3   A1      = 4.394989E-5    A2      = 0.6320223
+RDSW    = 4E3            PRWG    = -0.2146377     PRWB    = 0.1688991
+WR      = 1              WINT    = 1.569174E-7    LINT    = 0
+DWG     = -2.578547E-8   DWB     = 9.89001E-9     VOFF    = -0.1219424
+NFACTOR = 1.8063821      CIT     = 0              CDSC    = 2.4E-4
+CDSCD   = 0              CDSCB   = 0              ETA0    = 0.0518465
+ETAB    = 4.735705E-3    DSUB    = 0.4254421      PCLM    = 2.7235598
+PDIBLC1 = 0              PDIBLC2 = 4.344554E-3    PDIBLCB = 4.528856E-3
+DROUT   = 5.604876E-3    PSCBE1  = 8E10           PSCBE2  = 5.04016E-10
+PVAG    = 4.6592572      DELTA   = 0.01           RSH     = 2.8
+MOBMOD  = 1              PRT     = 0              UTE     = -1.5
+KT1     = -0.11          KT1L    = 0              KT2     = 0.022
+UA1     = 4.31E-9        UB1     = -7.61E-18      UC1     = -5.6E-11
+AT      = 3.3E4          WL      = 0              WLN     = 1
+WW      = 0              WWN     = 1              WWL     = 0
+LL      = 0              LLN     = 1              LW      = 0
+LWN     = 1              LWL     = 0              CAPMOD  = 2
+XPART   = 0.5            CGDO    = 3.12E-10       CGSO    = 3.12E-10
+CGBO    = 1E-12          CJ      = 9.916255E-4    PB      = 0.8896731
+MJ      = 0.392439       CJSW    = 2.91776E-10    PBSW    = 0.99
+MJSW    = 0.1676363      CJSWG   = 4.42E-11       PBSWG   = 0.99
+MJSWG   = 0.1676363      CF      = 0              PVTH0   = 0.0107102
+PRDSW   = -233.4720278   PK2     = 1.861393E-3    WKETA   = -6.345721E-3
+LKETA   = -0.0207051      )
```