Parçacık Sürü Optimizasyonu Tabanlı Evirici Tasarımı
Inverter Design Based on Particle Swarm Optimization

Ozan Der, Revna Acar Vural, Tülay Yıldırım

Elektronik ve Haberleşme Müh. Bölümü
Yıldız Teknik Üniversitesi
der_ozan@hotmail.com, racar@yildiz.edu.tr, tulay@yildiz.edu.tr

Özet

Parçacık Sürü Optimizasyonu (PSO) son yıllarda kullanılan en yeni akıllı hesap yöntemlerinden biridir. PSO, karmaşık denklem takımı içeren lineer olmayan problemlerde başarı ile kullanılmaktadır. Klasik optimizasyon yöntemlerinden en önemli farkı türev kullanmamasıdır. Bu da sonuca daha kısa sürede ulaşmasını sağlamaktadır. Bu çalışmada Parçacık Sürü Optimizasyonu algoritmasının elektronik devre tasarımında kullanılabilirliği araştırılmış; bu amaçla bir evirici yapısı üzerinde algoritmanın başarısı denenmiştir. Tasarımı yapılacak olan eviricinin performans kriterleri, PSO algoritmasının sınırlamalarını oluşturmaktadır. Elde edilen sonuçlar eviricinin teorik hesaplamalarla tasarımının PSO tabanlı tasarımı ile uyumlu olduğunu göstermiştir.

Abstract

Particle Swarm Optimization (PSO) is one of the novel intelligent computational methods. PSO is successfully used in nonlinear problems which contain complex equations. The most important difference of PSO than the traditional optimization techniques is having no derivation operation which leads to shorter computation time. In this work, usage of Particle Swarm Optimization algorithm in electronic circuit design has been investigated. For this purpose, the performance of the algorithm has been tested on the design of an inverter. Performance criteria of inverter constitute the constraints of PSO. Obtained results show that theorical design of inverter is matched with PSO based design.

1. Giriş

Parçacık Sürü Optimizasyonu (PSO), 1995 yılında Kennedy ve Eberhart tarafından kuş ve balık sürülerinin iki boyutlu davranışlarından esinlenilerek geliştirilmiş populasyon tabanlı bir optimizasyon tekniğidir [1]. Bu iki boyutlu davranışlar; çevrelerine adapte olabilme, zengin yiyecek kaynakları bulabilme ve avcılardan kaçabilme gibi 'bilgi paylaşma' yaklaşımı gerektirmektedir. Bilgi paylaşımı sayesinde sürüler, yiyecek ararken ya da avcıdan kaçarken, hedefe en yakın olan sürü elemanını takip ederler ve kendi hızlarını ve konumlarını en başarılı elemana göre güncellerler.

PSO optimum ya da optimuma yakın çözüm bulmak için önce her biri çözüm adayı olan parçacıklar oluşturur. Bu bireyler belli sınırlar içerisinde rasgele seçilir. Bireylerin bir araya gelmesiyle çözüm için gerçekleştirilen populasyon oluşturulur. Parçacık hareket ettiğinde koordinatlarını bir fonksiyona gönderir ve parçacığın uygunluk değeri (optimum çözüme olan uzaklığı) ölçülmüş olur. Parçacığın konum bilgisi (koordinatlarını), hızı (çözüm uzayında ne kadar hızla ilerlediği) ve güncel en iyi uygunluk değeri ile bu değeri elde ettiği koordinatları hafızada tutulmalıdır. Çözüm uzayındaki her boyutta hızının ve yönünün nasıl değişeceği, komşularının en iyi koordinatları ve kendi kişisel en iyi koordinatlarının bir birleşimi olacaktır. [2]

PSO literatürde çok çeşitli uygulama alanlarında başarıyla kullanılmıştır. [3]'te PSO tabanlı bir imge bölütleme yaklaşımı sunulmuştur. Gauss olasılık fonksiyonlarından oluşan bir karışım kullanılarak verilmiş bir görüntünün histogramı uydurulmuş ve PSO algoritması bu olasılık fonksiyonları karışımının parametrelerini hesaplamıştır. Böylelikle yoğunluk fonksiyonu ile gerçek histogram arasındaki hata en aza düşürülmüştür. [4]'te PSO, sınırlamalar içeren bazı mekanik tasarım problemlerine uygulanmıştır. Bu problemlerde minimize edilecek fonksiyonun yanı sıra fonksiyonların parametrelerini sınırlayan başka denklem takımları da mevcuttur. Hayli karmaşık bu problemlerin PSO ile çözümü diğer yöntemlerle karşılaştırılmış ve PSO'nun en avantajlı ve başarılı yöntem olduğu saptanmıştır. [5]'te PSO, güç sistemlerinde, lineer olmayan optimizasyon problemi olarak tanımlanmış reaktif güç ve gerilim kontrol (Voltage/VAR control) problemine uygulanmıştır. Burada minimize edilecek durum, gerilim kararlılığının da sağlanması kaydıyla hedef güç sistemindeki toplam güç kaybıdır. [6]'da PSO, FPGA yerleştirme ve yönlendirmesinde kullanılmıştır. Algoritma, yapılandırılabilir lojik bloklar (CLB) arasındaki bağlantı uzunluklarının minimize edilmesini sağlamıştır. [7]'de PSO, şebekeden bağımsız fotovoltaik sistem için tasarlanan bulanık lojik denetleyicisinin optimizasyonunda kullanılmıştır. PSO, denetleyicinin tasarımında kullanılan üyelik fonksiyonunu ve kural kümesini optimize etmiştir. Başka bir çalışmada PSO, devre ayrıştırma problemine uygulanmış ve ayrıştırma yaklaşımının parametreleri PSO tarafından belirlenmiştir [8]. Burada PSO tabanlı yaklaşım, test vektörlerinin sayısını, ayrıştırma sayısını ve kritik hattı optimize etmiştir. [9]'da PSO, hareketli sensörler veya robotlar sürüsünün haberleşmesinde; yangın söndürme, mayın bulma ve radyoaktivitenin belirlenmesi gibi yerel ve küresel işlerin azami uygulanması için kullanılmıştır. [10]'da PSO ve evrimsel algoritma kullanılarak hibrid bir yapı oluşturulmuş ve bu yapı kombinezonsal lojik devrelerin evrimleşmesinde kullanılmıştır. Bu çalışmada amaç %100 işlevsel kombinezonsal lojik devre evrimini minimum kapı sayısı ile

yapmaktır. Çalışmanın sonunda hibrid yapının bu amacı başarıyla gerçekleştirdiği belirtilmektedir.

Bu çalışmada literatürde yapılanlardan farklı olarak PSO algoritmasının elektronik devre tasarımında kullanılabilirliği araştırılmış; bu amaçla bir evirici yapısı üzerinde algoritmanın başarısı denenmiştir. Tasarımı yapılacak olan eviricinin performans kriterleri, 2.bölümde anlatılan algoritmanın sınırlamalarını oluşturmuştur. 3. bölümde eviricinin tasarımından bahsedilmiştir. Eviricinin çıkış düşme zamanı denklemi, bir takım düzenlemeler yapıldıktan sonra algoritmanın uygunluk fonksiyonu olarak seçilmiş ve bu fonksiyonun 0'a çok yakın bir değer alması sağlanana kadar algoritma koşturulmuştur. 4. bölümde PSO algoritmasından elde edilen sonuçlar ve PSPICE sonuçları tablo halinde verilmiş, 5. bölümde ise bu sonuçlar tartışılmıştır.

2. Parçacık Sürü Optimizasyonu

PSO bir grup rasgele çözüm kümesi (parçacık sürüsü) ile başlar ve güncellemelerle optimum çözüm bulunmaya çalışılır. Her iterasyonda parçacık konumları, hafızada saklanacak iki en iyi değere (pbest ve gbest) göre güncellenir. Parçacığın elde ettiği en iyi çözümü sağlayan koordinatlar *pbest,* populasyonda tüm parçacıklar için o ana kadar elde edilen en iyi çözümü sağlayan koordinatlar ise gbest olarak adlandırılır.

Parçacık populasyon matrisinde D adet özelliğe sahip i. parçacık $x_i=[x_{i1},x_{i2},...,x_{iD}]$ şeklinde gösterilir. Her iterasyon için *pbest* ve *gbest* bulunduktan sonra parçacığın konumu ve hızı (1) ve (2)'ye göre güncellenir.

$$v_i^{k+1} = w.v_i^k + c_1.rand_1^k.(pbest_i^k - x_i^k) + c_2.rand_2^k.(gbest^k - x_i^k) \quad (1)$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (2)$$

Görüldüğü üzere parçacık sürü algoritmasındaki bu güncellemeler basit toplam ve çarpımlardan oluşmakta ve türev bilgisi gerektirmemektedir. Öğrenme faktörleri olan $\mathbf{c_1}$ **ve** $\mathbf{c_2}$ her parçacığı *pbest* ve *gbest* değerlerine doğru çeker. Denklemdeki $\mathbf{rand_1}$ ve $\mathbf{rand_2}$ ise 0-1 arasında seçilmiş rasgele sayılardır. **k** iterasyon sayısını göstermektedir. Eylemsizlik ağırlığı olan **w** her iterasyonda doğrusal olarak azaltılmalı ve birden küçük seçilmelidir. [2]

```
Rasgele parçacık kümesi oluştur
Konum ve hız için balşangıç koşullaını ve varsa sınır değerleri tanımla
   for epoch=1:max_epoch  %% max_epoch=maksimum iterasyon sayısı
      for i=1:n           %% n= parçacık sayısı
         for j=1:D        %% D= problemin boyutu
            konum denklemini güncelle
            konum bilgisinin sınır değerlerde kalmasını sağla
             if f( x i,j ) < costval  %% costval=değer fonksiyonu (eşik hata bilgisi)
               pbest'i güncelle
             end
               costval = f( x i,j )
         end
      end
      costval'in minimum değerini ve bu değeri aldığı satır bilgisini sakla

      for i=1:n
         for j=1:D
            hız denklemini güncelle
         end
      end
   end
```

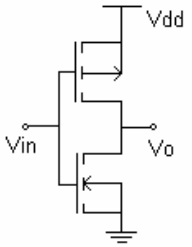
Şekil 1. PSO algoritmasının temel adımları

3. Evirici Tasarımı

PSO algoritmasının elektronik devre tasarımında kullanılabilirliğinin araştırılması için sayısal bir devre yapısı seçilmiş ve tasarım kriterleri programa tanıtılarak istenen sınırlar içerisinde devrenin tasarımının PSO algoritması tarafından yapılması hedeflenmiştir. Algoritma sonuç olarak sayısal devrenin içindeki MOS yapıların minimum W/L oranlarını ve belli sınırlar içinde verilen tasarım kriterlerinin tam değerlerini vermektedir. Burada tasarım problemini PSO'ya tanıtabilmek için tasarım kriterlerini ve MOS yapıların W/L oranlarını değişken olarak içeren bir denklem oluşturulmalıdır. PSO'nun bulması gereken, bu denklemin minimum noktası ve bu minimum noktayı sağlayan tasarım kriter değerleri ile W/L oranlarıdır.

3.1. Problemin tanımı

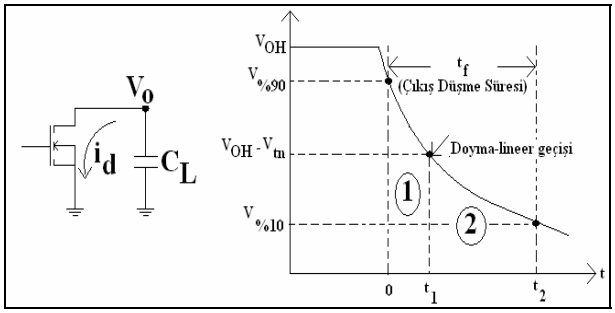
Aşağıda verilen evirici yapısında eşit geometriye sahip MOS yapıları bulunmaktadır. Yapılacak tasarımda çıkış düşme süresinin, çıkışa bağlanacak kapasite değerinin ve MOS yapıların W/L oranının belli değerler arasında tutulması istenmektedir. Bunun için öncelikle PSO algoritmasının minimize edeceği çıkış düşme süresini, W/L oranını ve kapasite değerini içeren denklemin elde edilmesi gerekmektedir.



Şekil 2. Evirici Yapısı

3.2. Çıkış düşme süresinin elde edilmesi

Eviricinin çıkış düşme süresini elde etmek için aşağıdaki şekilde gösterildiği üzere çıkışa bağlanan kapasitenin NMOS yapı üzerinden deşarj edilmesi gerekir.



Şekil.3 (a) Çıkışa bağlanan kapasitenin NMOS üzerinden deşarjının gösterimi (b) Çıkış geriliminin zamana bağlı değişimi üzerinde düşme süresinin gösterimi [11]

Şekil 3.b'de görüldüğü gibi çıkış düşme zamanı NMOS yapının t_1 anına kadar doymada, t_1- t_2 aralığında ise lineerde

çalıştığı göz önünde bulundurularak hesaplanmıştır. Buna göre toplam çıkış düşme zamanı olan t_2 (3)'de verilmiştir. [11]

$$t_f=(t_2-t_1)+t_1=\frac{C_L}{\beta_n(V_{DD}-V_{tn})}*\left[\frac{2(0.1V_{DD}-V_{tn})}{(V_{DD}-V_{tn})}+\ln\left(\frac{2(V_{DD}-V_{tn})-0.1V_{DD}}{0.1V_{DD}}\right)\right] \quad (3)$$

$$\beta_n=\mu C_{ox}\left(\frac{W}{L}\right)_n \quad (4)$$

(4) eşitliğini kullanarak ve yukarıdaki denklemi sağ tarafı 0'a eşitlenecek şekilde düzenlersek PSO'da kullanılacak olan değer fonksiyonunu elde etmiş oluruz.

$$(\mu C_{ox})*\left(\frac{W}{L}\right)_n*t_f-\frac{C_L}{(V_{DD}-V_{tn})}*\left[\frac{2(0.1V_{DD}-V_{tn})}{(V_{DD}-V_{tn})}+\ln\left(\frac{2(V_{DD}-V_{tn})-0.1V_{DD}}{0.1V_{DD}}\right)\right]=0 \quad (5)$$

(5) no.lu eşitliğin sol tarafı PSO algoritmasının uygunluk fonksiyonu olacaktır. Burada amaç, uygunluk fonksiyonunu 0'a çok yakın bir değere eşitleyecek değişkenlerin yani W/L oranı ile tasarım kriterlerine ait değerlerin bulunabilmesidir. Uygunluk fonksiyonunu sıfıra eşitlenmesi hatanın da sıfır olması demektir. Tablo 1'de bu denklemin bileşenlerinin PSO algoritmasında nasıl kullanılacağı gösterilmiştir.

Tablo. 1 PSO'da kullanılacak denklem bileşenleri bilgisi

Denklem Bileşenleri	Değerleri	PSO için giriş/çıkış
$\mu.C_{ox}$	Kullanıcının belirleyeceği bu değerler tasarımda kullanılan teknoloji parametrelerine bağlıdır.	GİRİŞ
V_{dd}		
V_{tn}		
W/L	Sadece değer aralığı verilecek, tam değeri PSO bulacaktır.	ÇIKIŞ
t_f		
C		

Eviricinin tasarımı için TSMC 0.25u parametreleri kullanıldığı için Vdd=2.5V, Vtn=0.3655V, $\mu.C_{ox}$ = 243.6 uA/V^2 olarak alınmıştır. Farklı üretim teknolojileri için bu değerler de değişecektir.

4. Sonuçlar

Bu uygulamada başlangıç populasyonu için 10 adet parçacık seçilmiş olup parçacık küme boyutu 10x3'tür. Başlangıç konumları W/L, C_L ve t_f için ayrı ayrı belirlenen sınır değerleri arasından rasgele verilmiştir. Hız güncellemesinde kullanılan c_1, c_2 ve w değerleri sırasıyla 2, 2 ve 0.99 olarak alınmıştır. Maksimum hata değeri 0.000001 olacak şekilde seçilmiş ve tüm tasarım örnekleri için algoritma sonucu bulunan değerlerin yerine konulduğunda uygunluk fonksiyonunun belirlenen hata değerinden daha küçük sonuçlar verdiği görülmüştür. İterasyon sayısının en üst değeri 100 olarak seçilmiştir. Eğer maksimum iterasyon sayısına ulaşmadan, algoritma hatayı minimuma indirecek küresel çözümü (gbest) bulduysa, döngüden çıkılmış ve işlem yükü hafifletilmiştir.

Bu çalışmada PSO algoritması kullanılarak 9 farklı evirici tasarımı gerçekleştirilmiştir. Her tasarım için belirlenen kriterlerin hangi aralıkta olduğu Tablo 2'de verilmiştir.

Tablo. 2 Her tasarım için belirlenen tasarım kriterleri

Tasarım	NMOS boyutu (W/L)	Çıkış düşme zamanı (t_f-ns)	Kapasite (C_l-pF)
1	1–2	4–7	0.5–1,5
2	2- 3	5- 7	0.5- 2,5
3	0.5- 2,5	3- 5	0,5- 1
4	1- 3	5- 8	2- 3
5	**1,5- 2,5**	**3.5- 7**	**1- 2,5**
6	3- 5	3- 7,5	2- 5
7	2- 4	4- 8	1- 4
8	1- 3	4- 6,5	0.4- 1
9	1- 2	5- 6,5	1- 2

Daha önce de bahsedildiği üzere Parçacık Sürü Algoritması ilk olarak rasgele bir populasyon oluşturur ve bu populasyondaki her parçacık birer çözüm adayıdır. Bu parçacıklar küresel çözüme ne kadar uzak olurlarsa olsun, kendi kişisel en iyi koordinatları ve komşularının en iyi koordinatlarının doğrultusunda algoritmadaki hız ve konum güncellemeleri ile küresel çözümün etrafında toplanmaktadır. Örnek olarak 5.tasarıma ait başlangıç populasyonu ve her parçacık için algoritma sonucu elde edilen küresel değer Tablo 3'te verilmiştir.

Tablo. 3 5.tasarıma ait başlangıç kümesi ve küresel çözüm

Başlangıç Populasyonu			PSO sonucu (küresel çözüm)		
W/L	t_f(ns)	C_l (pF)	W/L	t_f(ns)	C_l (pF)
10	241	67	1,6	5,922	1,846
136	123	144	1,6	5,922	1,846
4514	16	142	1,6	5,922	1,846
15	82	17	1,6	5,922	1,846
176	14	517	1,6	5,922	1,846
17	15	25	1,6	5,922	1,846
184	93	12	1,6	5,922	1,846
13455	91	3	1,6	5,922	1,846
13	12	5	1,6	5,922	1,846
1932	15	112	1,6	5,922	1,846

5. tasarım için belirlenen tasarım kriterleri (1,5< W/L < 2,5; 3,5< t_f <7; 1< C_l <2,5.) PSO algoritmasının sınırlamalarını oluşturmaktadır. Küresel değer maksimum iterasyon sayısından önce, 72. adımda bulunmuştur ve döngüden çıkılmıştır. Bu örnek için tüm parçacıkların küresel değer etrafında toplandığı görülmektedir. Ancak küresel çözüme ulaşmak için Ancak en iyi çözüme ulaşmak için tüm parçacıkların bir araya toplanmasını beklemek gerekmez. Her iterasyon sonucunda bulunan "gbest" değeri istediğimiz uygunluk fonksiyonunu maksimum hata farkıyla sağlıyorsa bu "gbest" değerini bulan parçacık çözüm olarak alınabilir.

PSO algoritması, Tablo 2'de tasarım kriterleri verilen 9 farklı evirici tasarımı için çalıştırılmış ve her tasarım için bulunan *gbest* değerleri (NMOS boyutu, çıkış düşme zamanı, kapasite değeri) sonuç olarak Tablo 4'te verilmiştir.

Tablo. 4 Her tasarım için PSO tarafından belirlenen değerler

Tasarım	**NMOS boyutu (W/L)**	**Çıkış düşme zamanı (t_f-ns)**	**Kapasite (C_l-pF)**
1	1,64	4,3	1,38
2	2,18	5,17	2,19
3	0,96	3,95	0,74
4	1,91	6,16	2,29
5	1,6	5,92	1,85
6	3,83	4,28	3,19
7	2,79	5,16	2,81
8	1,04	4,73	0,96
9	1,31	5,35	1,37

Yukarıdaki tablodan da anlaşılacağı üzere evirici yapısı için PSO algoritması istenen tasarım kriterlerini sağlayacak minimum W/L oranını bulmuştur. Karşılaştırma yapmak amacıyla, PSO'nun bulduğu W/L oranını ve çıkışa bağlanacak kapasite değerini kullanarak PSPICE ortamında tasarlanan eviricilerin çıkış düşme zamanları Tablo 5'de verilmektedir.

Tablo. 5 PSO tarafından belirlenen NMOS boyut ve kapasite değerleri için PSPICE simülasyon sonuçları

	PSO sonuçları – PSPICE girdisi		**PSPICE sonucu**
Tasarım	**NMOS boyutu (W/L)**	**Kapasite (pF)**	**Çıkış düşme zamanı (t_f-ns)**
1	1,64	1,38	11,21
2	2,18	2,19	14,66
3	0,96	0,74	8,72
4	1,91	2,29	15,18
5	1,6	1,85	14,47
6	3,83	3,19	12,88
7	2,79	2,81	14,32
8	1,04	0,96	10,82
9	1,31	1,37	11,47

5. Tartışma

Parçacık Sürü Optimizasyonu son yıllarda kullanılan en yeni akıllı hesap yöntemlerinden biridir. PSO'nun yapısı karmaşık denklem takımı içeren lineer olmayan problemlerde başarı ile kullanılmaktadır. Klasik optimizasyon problemlerinden en önemli farkı türev kullanmamasıdır. Bu da sonuca daha kısa sürede ulaşmasını sağlamaktadır. Bu çalışmada basit bir evirici yapısı üzerinde PSO algoritmasının performansı denenmiştir. Burada teorik hesaplamalarda kullanılan denklem takımı [11] PSO algoritmasındaki uygunluk fonksiyonu olarak kullanılmış ve bu fonksiyonu minimuma çeken tasarım kriteri değerlerine başarı ile ulaşılmıştır. Elde edilen PSO tabanlı tasarım sonuçları PSPICE sonuçları ile de karşılaştırılmıştır. Aradaki farklılık PSPICE'ın teorikte kullanılandan daha karmaşık bir denklem takımı kullanarak hesaplama yapmasından kaynaklanmaktadır. İleriki çalışmalarda daha karmaşık elektronik devre yapılarının tasarımında PSO kullanımı araştırılacaktır.

6. Kaynaklar

[1] Kennedy, J. , Eberhart, R. C. "Particle swarm optimization" Proc. IEEE int'l conf on neural Networks, Vol. IV, pp. 1942–1948. Piscataway, NJ, 1995.

[2] Tamer, S. , Karakuzu C. "Parçacık Sürüsü Optimizasyon Algoritması ve Benzetim Örnekleri", ELECO 2006, Elektronik Bildirileri Kitabı, 2006.

[3] Chih-Chin Lai. "A Novel Image Segmentation Approach Based on Particle Swarm Optimization". IEICE Transactions 89-A(1): 324–327, 2006.

[4] X. Hu, R. Eberhart, Y. Shi."Engineering optimization with particle swarm", IEEE Swarm Intelligence Symposium, Indianapolis, USA, 2003.

[5] Yoshida, H., Kawata, K., Fukuyama, Y., Nakanishi, Y. " A particle swarm optimization for reactive power and voltage control considering voltage stability", Proc. IEEE Intl. Conf. on Intelligent System Application to Power Systems, Rio de Janeiro, Brazil, pp:117–121, 1999.

[6] Gudise, V. G. and Venayagamoorthy, G. K. "FPGA Placement and Routing Using Particle Swarm Optimization" IEEE Computer Society Annual Symposium on VLSI, USA, pp:307-308, February 2004,

[7] Welch, R. and Venayagamoorthy, G. K. "A Fuzzy-PSO Based Controller for a Grid Independent Photovoltaic System", IEEE Symposium on Computational Intelligence and Data Mining, USA, April 2007.

[8] Venayagamoorthy, G. K., Smith, S. C. and Singhal, G. " Particle Swarm Based Optimal Partitioning Algorithm for Combinational CMOS Circuits." Engineering Applications of Artificial Intelligence ,pp: 177-184,2007.

[9] Venayagamoorthy, G. K. and Doctor, S. "Navigation of Mobile Sensors Using PSO and Embedded PSO in a Fuzzy Logic Controller", 39th IEEE IAS Annual Meeting on Industry Applications, USA, pp:1200-1206, 2004.

[10] Gudise, V. G. and Venayagamoorthy, G. K. Evolving Digital Circuits Using Particle Swarm. INNS-IEEE International Joint Conference on Neural Networks, USA, pp: 468-472, July 2003.

[11] Kang K-M., Leblebici Y., "CMOS Digital Integrated Circuits: Analysis and Design", McGraw Hill, 1999.