İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

# STOKASTİK HESAPLAMA ALTERNATİFİ OLARAK BİT KATARI HESAPLAMA İLE HATASIZ ARİTMETİK İŞLEM BLOKLARININ TASARIMI

**YÜKSEK LİSANS TEZİ**

**Ensar VAHAPOĞLU**

**Elektronik ve Haberleşme Mühendisliği Anabilim Dalı**

**Elektronik Mühendisliği Programı**

**HAZİRAN 2018**

**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

# STOKASTİK HESAPLAMA ALTERNATİFİ OLARAK BİT KATARI HESAPLAMA İLE HATASIZ ARİTMETİK İŞLEM BLOKLARININ TASARIMI

**YÜKSEK LİSANS TEZİ**

**Ensar VAHAPOĞLU**
**(504151205)**

**Elektronik ve Haberleşme Mühendisliği Anabilim Dalı**

**Elektronik Mühendisliği Programı**

**Tez Danışmanı: Dr. Öğr. Üyesi Mustafa ALTUN**

**HAZİRAN 2018**

İTÜ, Fen Bilimleri Enstitüsü'nün 504151205 numaralı Yüksek Lisans Öğrencisi Ensar VAHAPOĞLU, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "STOKASTİK HESAPLAMA ALTERNATİFİ OLARAK BİT KATARI HESAPLAMA İLE HATASIZ ARİTMETİK İŞLEM BLOKLARININ TASARIMI" başlıklı tezini aşağıdaki imzaları olan jüri önünde başarı ile sunmuştur.

| | | |
|---|---|---|
| **Tez Danışmanı :** | **Dr. Öğr. Üyesi Mustafa ALTUN**<br>İstanbul Teknik Üniversitesi | ..... |
| **Jüri Üyeleri :** | **Doç. Dr. Sıddıka Berna ÖRS YALÇIN**<br>İstanbul Teknik Üniversitesi | ..... |
| | **Prof. Dr. Günhan DÜNDAR**<br>Boğaziçi Üniversitesi | ..... |
| | | ..... |

**Teslim Tarihi :** **2 Mayıs 2018**
**Savunma Tarihi :** **4 Haziran 2018**

***Eşime ve Aileme,***

## ÖNSÖZ

Bu çalışmanın olgunlaşmasında büyük katkısı olan tez danışmanım Dr. Öğr. Üyesi Mustafa ALTUN'a, aydınlatıcı fikirleriyle bakış açımı genişleten Dr. Öğr. Üyesi İsmail ÇEVİK ve Dr. Tuba AYHAN'a, yüksek lisans öğrenim sürecimin keyifli geçmesini sağlayan Nanoelektronik ve Hesaplama Grubu'ndaki bütün çalışma arkadaşlarıma teşekkürü bir borç bilirim.

Haziran 2018

Ensar VAHAPOĞLU
(Araştırma Görevlisi)

**İÇİNDEKİLER**

**Sayfa**

## KISALTMALAR

**SH** : Stokastik Hesaplama
**BKH** : Bit Katarı Hesaplama
**LFSR** : Doğrusal Geribesleme Kaydırma Yazmacı
**AAKT** : Asenkron Artan Katar-uzunluklu Toplayıcı
**AAKÇ** : Asenkron Artan Katar-uzunluklu Çarpıcı
**SAKT** : Senkron Artan Katar-uzunluklu Toplayıcı
**SAKÇ** : Senkron Artan Katar-uzunluklu Çarpıcı
**SSKT** : Senkron Sabit Katar-uzunluklu Toplayıcı
**SSKÇ** : Senkron Sabit Katar-uzunluklu Çarpıcı
**TS** : Transistor Sayısı
**YO** : Yanlış Tasnif Oranı

**SEMBOLLER**

| | |
|---|---|
| **n** | **:** Bit Katarı Uzunluğu |
| **m** | **:** Çıkış Bit Katarı Uzunluğu (Ardışık Bit Katarlarının İşlenmesi Sırasında) |
| **W** | **:** Bit Genişliği |
| $\mathbf{z_e}$ | **:** Çıkış katarının beklenen değeri |
| **W** | **:** Bit Genişliği |

## ÇİZELGE LİSTESİ

**Sayfa**

## ŞEKİL LİSTESİ

**Sayfa**

# STOKASTİK HESAPLAMA ALTERNATİFİ OLARAK BİT KATARI HESAPLAMA İLE HATASIZ ARİTMETİK İŞLEM BLOKLARININ TASARIMI

## ÖZET


Stokastik hesaplama (SH), seri bit katarlarını olasılıksal olarak işleyen bir hesaplama paradigmasıdır. Aslen 1950'lerde teklif edilmesine karşın, halen kullanılan ikili hesaplamaya göre doğruluk konusunda geride kaldığı için gözden düşmüştür.

Son yıllarda entegre devre tasarımının fiziksel sınırlarına yaklaşmasıyla birlikte, hatırı sayılır derecede alan tasarrufu sağlayan stokastik hesaplama yeniden gündeme gelmiştir. Özellikle kesin doğruluk gerektirmeyen (işaret işleme, yapay sinir ağları vb.) uygulamalarda sıklıkla kullanılmaya başlamıştır. Ancak, doğruluk problemi stokastik hesaplamanın kullanımını sınırlamaya devam etmektedir.

Bu çalışmada, stokastik hesaplama ile aynı mantığı kullanan "Bit Katarı Hesaplama" adlı yeni bir hesaplama paradigması teklif edilmiştir. Bit Katarı Hesaplama, temel olarak stokastik hesaplamanın alan avantajı ile geleneksel ikili hesaplamanın doğruluk avantajını birleştirir. Stokastik hesaplamadaki gibi rastgele veya Binom dağılımlı katarlar kullanılmasına ihtiyaç duymaz. Geleneksel ikili hesaplamadaki gibi bilgiyi paralel bitlerle işlemek yerine seri bit katarları kullanarak devrelerin alan maliyetini önemli derecede düşürür. Neticede, bu paradigma ile üretilen tasarımlar düşük alan maliyeti, yüksek doğruluk gibi modern ihtiyaçlara cevap verebilecek kapasitededir.

Geleneksel hesaplamanın temel aritmetik işlem blokları toplayıcı ve çarpıcı devrelerdir. Bu çalışmada da Bit Katarı Hesaplama ile toplayıcı ve çarpıcı devreler teklif edilmiştir. Ancak, yapılan tasarımların stokastik veya bit katarı hesaplama ile elde edilebilecek optimum sonuçlara ne kadar yakın olduğunu anlamak amacıyla, öncelikle bu paradigmaların giriş/çıkış bit katarı uzunluğu, doğruluk, devre karmaşıklığı gibi ölçütler açısından sınırları belirlenmiştir.

Farklı ihtiyaçlar için seçenekleri arttırmak amacıyla hem asenkron hem de senkron tasarımlara yer verilmiştir. Ayrıca ardışık katarları işleyebilen tasarımlar da teklif edilmiştir. Bütün tasarımların doğruluk ve alan karmaşıklığı sonuçları, stokastik/bit katarı hesaplama ile elde edilebilecek optimum sınırlara oldukça yakındır.

Bit katarı hesaplama kullanılarak teklif edilen toplayıcı ve çarpıcı devreler, geleneksel ikili veya stokastik hesaplama ile tasarlanan literatürdeki benzerleri ile alan, gecikme ve doğruluk yönünden karşılaştırılmıştır. Bu çalışmanın özgün yanlarından bir diğeri, karşılaştırma için yapılan benzetimlerin transistör seviyesinde olmasıdır. Böylece kapı veya sistem seviyelerindeki benzetimlerde ortaya çıkmayan zamanlama problemleri de incelenmiştir.

Benzetimler Cadence ortamında AMS 0.35µm CMOS teknolojisinde transistör ve kapı seviyesinde yapılmıştır. Yapılan karşılaştırmalarda, teklif edilen toplayıcı ve çarpıcı devrelerin stokastik benzerlerine her açıdan üstünlük sağladığı görülmüştür. Geleneksel ikili devreler ise sadece gecikme konusunda geride bırakılamamıştır ki,

gecikme seri katarlar kullanan herhangi bir devre için katlanılması gereken bir sorundur.

Karşılaştırmalar ayrıca basit bir yapay sinir ağları uygulamasında da yapılmıştır. Bu uygulamada da, teklif edilen devrelerin doğruluktan neredeyse hiç feragat etmeyerek büyük ölçüde alan tasarrufu yaptığı görülmüştür.

Sonuç olarak, bu çalışma ile geleneksel ikili veya stokastik hesaplama ile üretilenlere göre çok daha küçük ve yüksek doğruluklu aritmetik işlemler gerçekleyebilen hesaplamalar için yeni ufuklar açıldığına inanılmaktadır.

**DESIGN OF ACCURATE ARITHMETIC OPERATION BLOCKS VIA BIT STREAM COMPUTING AS AN ALTERNATIVE TO STOCHASTIC COMPUTING**

## SUMMARY

Stochastic computing is a computing paradigm that process serial bit streams probabilistically. It uses randomly generated streams and stores the information in them as probability of 1s. Despite its initial emergence at 1950s, it loses favor since it got behind in still-used binary computing in terms of accuracy.

Recently, mostly because integrated circuit design has been approaching its physical limits, stochastic computing, providing substantial amount of area savings, has been brought to agenda. It started to be used frequently especially on the applications not requiring perfect accuracy like signal processing, artificial neural networks, etc. Nevertheless, accuracy problem still continues to limit stochastic computing.

In this work, a new computing paradigm “Bit Stream Computing" that is constructed on the logic used in stochastic computing has been proposed. Bit Stream Computing can be considered as a generalized version of stochastic computing. It mainly combines the area advantage of stochastic computing with accuracy advantage of conventional binary computing. It does not need to employ randomly generated or Binomially distributed streams. It decreases the area cost of the circuits by processing the information with serial bits instead of using parallel bits as in conventional binary computing. As a result, the circuits produced via this paradigm has the potential to satisfy the current needs such as low cost and high accuracy.

A computing paradigm primarily needs arithmetic operations. The main arithmetic operation blocks of conventional computing are addition and multiplication circuits. Likewise, this work proposes addition and multiplication circuits with bit stream computing to lay a foundation. However, as a beginning, it is preferred to determine the limits of stochastic or bit stream computing in terms of the metrics like input/output stream length, accuracy, circuit complexity. This determination facilitates to check any proposed design whether its results are close to optimum ones.

The determination of the limits of stochastic/bit stream computing also reveals some interesting outcomes:

- Randomly generated or binomially distributed streams are not be able to produce 100% accurate results, i.e. perfect accuracy needs deterministic streams.
- Addition and multiplication operations also require longer output streams than of inputs for perfect accuracy.
- 100% accurate stochastic/bit stream computing circuits are not compatible with successive stream processing.

These outcomes have given reasons to branch the circuits proposed with bit stream computing. Both synchronous and asynchronous designs are proposed to satisfy various requirements. For instance, generally asynchronous designs are more cost effective than synchronous ones in cases of shorter bit streams. Synchronous designs are also classified as two subgroups named constant stream and increased stream designs. The former group is suitable for successive stream processing while the latter has %100 accuracy. The proposed asynchronous designs are also perfectly accurate, therefore they are not suitable for successive stream processing. The reason why any asynchronous design that suitable for successive stream processing is not proposed has also been explained. The accuracy and area complexity results of all proposed designs are quite close to the optimum limits which can be achieved via stochastic/bit stream computing.

Adder and multiplier circuits proposed with bit stream computing are compared with their conventional binary and stochastic predecessors in the literature in terms of area, delay, and accuracy. One of the distinctive features of this work is its that simulations carried out for comparisons are in transistor level. Nearly all previous works in stochastic computing do not have any transistor level simulations which results in to overlook some drawbacks that can be only noticed in this simulations such as time alignment problems. On the other hand, the proposed circuits and their predecessors are also examined in this simulations and they are tried to be optimized to get rid of this unnoticed problems.

Simulations are carried out in both transistor and gate level with AMS 0 35µm CMOS technology in Cadence environments. Accuracy comparisons with transistor level simulations are made with two novel metrics suitable for stochastic/bit stream computing: integrity and correctness. These metrics pave the way to examine any stochastic/bit stream computing design with regards to the drawbacks coming to existence in transistor level simulations.

Delay comparisons are also made by dividing the delay results into two metrics: inherent circuit delay and stream delay. The results have shown that the proposed circuits outperform any former stochastic design. Another outcome of these comparisons is that the delay amounts stochastic/bit stream computing designs are mainly comes from stream delay instead of inherent circuit delay so that considering only stream delay for these designs is a quite rational simplification.

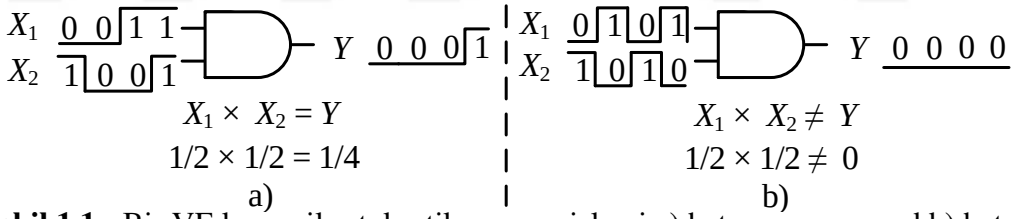
Area comparisons prove that our proposed designs are most area-efficient ones among all compared designs. One of them (Synchronous Constant Stream Adder) has even $O(1)$ complexity with respect to bit lengths, i.e. it is completely scalable. With zero or tiny sacrifice from accuracy, proposed designs provide substantial amount of area savings. Out of these designs, asynchronous ones have low cost for smaller bit lengths, while synchronous ones are more area-friendly for longer.

Lastly, a simple neural network is implemented with all compared designs to show our designs' supremacy in an application. Comparisons are also classified with regard to input/output signal format (analog, binary and bit stream). The implementations are compared in terms of area (number of transistors) and accuracy (misclassification rate). Our perfectly accurate designs generally have better area performance than other binary or stochastic designs. On the other hand, the area results of our semi-accurate (constant stream/suitable for successive processing) designs have the lowest area cost in exchange for an insignificant accuracy lost.

To sum up, this work proposes a novel computing paradigm “Bit Stream Computing" which has the ability to realize the main arithmetic operations more efficiently than previous computing paradigms. It is believed that this work opens up new horizons for computing paradigms that is capable of realizing much smaller and more accurate arithmetic operations than conventional binary or stochastic computing.

# 1. GİRİŞ

İlk olarak 1960'lı yıllarda ortaya çıkan Stokastik Hesaplama (SH) [1, 2], binom dağılımlı bit katarlarını seri olarak işleyen bir hesaplama türüdür. Her bit katarı bir stokastik değeri temsil eder. Bu stokastik değer, bit katarındaki 1 değerli bitlerin sayısının toplam bit sayısına bölümü ile elde edilir. Böylece, $n$ bitlik katarla gösterilen bir giriş veya çıkış $0/n$'den $n/n$'e kadar $n+1$ farklı değer temsil edilebilir ki, geleneksel bir ikili giriş/çıkış düğümü sadece 0 veya 1 değerine sahip olabilir. Stokastik hesaplamanın bu özelliği özellikle aritmetik operasyonların gerçeklenmesi sırasında önemli avantaj sağlar. Örneğin, çarpma işlemi Şekil 1.1'de görüldüğü gibi bir VE kapısı ile gerçeklenebilir. Bu şekilde 1/2 değerine sahip iki bit katarı VE kapısı ile çarpılmıştır. Şekil 1.1.a)'daki işlem çıkışta beklenilen 1/4 değerini verir ancak katarlardaki 1 ve 0 bitlerinin farklı bir dağılımı doğru sonucun elde edilmemesine sebep olabilir (Şekil 1.1.b)).

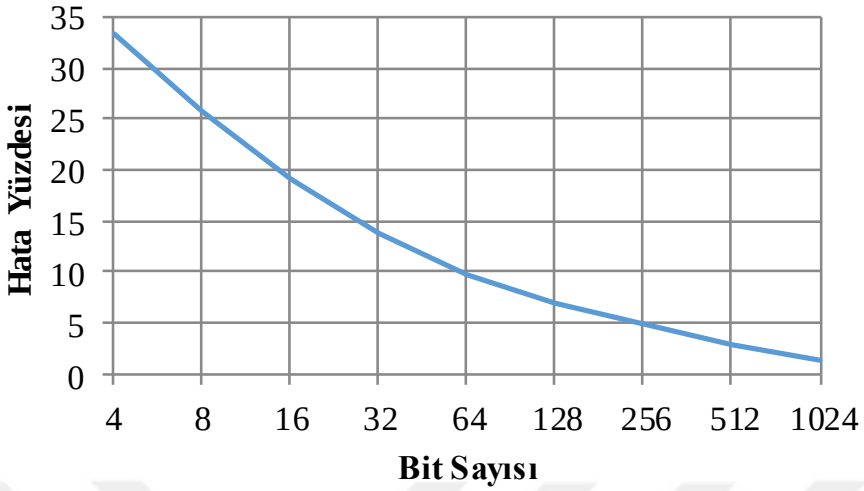



**Şekil 1.1** : Bir VE kapısı ile stokastik çarpma işlemi a) hatasız sonuç, and b) hatalı sonuç.

## 1.1 Tezin Amacı

Giriş katarlarının binom dağılımlı oluşu, SH'da sıfır hata veya standart sapma elde etmenin ancak sonsuz uzunlukta katarlar ile elde edilebileceği anlamına gelmektedir. Diğer bir deyişle, hatasız SH imkansızdır. Şekil 1.2 değişen bit katarı uzunluklarına göre iki girişi de 1/2 değerli olan VE kapısının ortalama hatasının değişimini göstermektedir. %10 veya %1 hatanın altına düşmek için bile sırasıyla 100 veya 1000 bitlik katarlarının kullanılması gerekir ki, bu uzunluktaki katarlar pratik olmayan hesaplama sürelerine sebep olurlar. Bu durum çok ciddi bir alan avantajı olmasına rağmen SH'nin geleneksel hesaplamaya neden rakip olamamasının da ana sebebidir

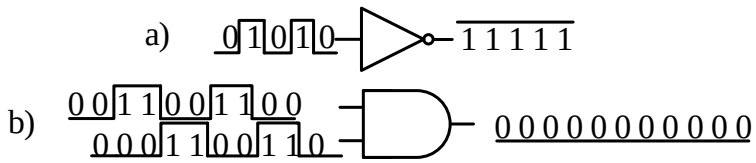
[3]. Yüksek hata oranları ve uzun hesaplama süreleri SH'nın önündeki en büyük engellerdir.




**Şekil 1.2** : Katarlardaki değişen bit sayılarına ($n$) göre bir VE kapısının ortalama hata oranları (iki girişin değeri de 1/2 iken).

Bu tez, SH mantığı üzerine inşa edilen, ancak rasgele veya binom dağılımlı bit katarları kullanmak zorunda olmayan "Bit Katarı Hesaplama (BKH)" adlı yeni bir hesaplama paradigması teklif etmeyi amaçlamaktadır. Bu paradigma ile yapılan hesaplamalarda stokastik veya deterministik katarlar kullanılabilecektir. BKH SH'nin alan avantajı ile geleneksel hesaplamanın doğruluk avantajını birleştirir. Bu tezde BKH kullanılarak yüksek doğruluklu çarpma ve toplama işlemleri teklif edilmiştir.

SH'nin doğumundan beri gündeminde olan doğruluk problemi kadar zamanlama problemleri de etkilidir. Zamanlama problemleri, genellikle bit katarındaki 1 ve 0'ların sürelerindeki istenmeyen sapmalardan dolayı meydana gelirler. Bu duruma iki örnek Şekil 1.3'de verilmiştir. Örneklerdeki katarların bir bit genişliği $X$'tir. Şekil 1.3.a)'daki problem bir lojik kapının gecikmesinin $X$'den büyük veya $X$'e eşit olamamasıdır ki, gerçekten de kullanılan teknolojiye bağlı olarak gecikmenin $X$'den çok küçük olması beklenir. Şekil 1.3.b) ise yanlış hizalama problemini göstermektedir; girişlerin kusursuz olarak hizalandığı durumda çıkışta iki adet 1 biti görülmesi beklenirken, hiç görülmemektedir. Bu tür zamanlama problemleri göz önüne alındığında, SH veya BKH ile geliştirilen herhangi bir devre tasarım tekniğinin zamanlama değerlendirmesi ile doğrulanması gerekir. Bu amaçla bu tezde teklif edilen bütün devreler transistör seviyesinde test edilmiştir.




**Şekil 1.3** : Bit süresi *X*'in değişimi için örnekler: a) *X* kadar gecikmesi olan evirici, b) Girişlerin hizalarının *X* kadar kayması.

## 1.2 Literatür Araştırması ve Tezin Katkıları

SH'de doğruluğu arttırmanın en yaygın çözümü giriş bit katarlarının rasgelelikleri azaltılarak veya birbirleri ile korelasyonlu hale getirilerek manipüle edilmeleridir. Bu amaçla sözde-rasgele veya yarı-rasgele rasgele sayı üreteçleri teklif edilmiştir [4, 5]. Sözde-rasgele üreteçler genellikle kusursuz doğruluk sağlayan 0 ve 1 sıralamaları üreten LFSR (Doğrusal Geribesleme Kaydırma Yazmacı-*Linear Feedback Shift Register*)'lar kullanırlar [5]. Ayrıca doğruluk için korelasyondan faydalanan [6] ve tamamen deterministik üreteçler kullanan [7, 8] çalışmalar da mevcuttur. [7] ve [8]'te kusursuz aritmetik işlemler elde edilmiştir. Ancak, bütün bu çalışmalarda her bir giriş için ayrı bit katarı üreteçlerine ihtiyaç duyulur. Bu yüzden üreteç sayısı giriş sayısına doğrusal olarak bağlıdır ve bu üreteçler devre alanının çoğunluğunu kaplarlar. [8]'de rasgele üreteçlere göre daha küçük deterministik üreteçler kullanılmasına rağmen, halen her girişin bir saat üretim devresi içeren kendi üreteci vardır.

Bahsi geçen çalışmaların bir diğer önemli mahzuru da çok seviyeli tasarımlara uygun olmamalarıdır. Bir seviyenin çıkışı direkt bir sonraki seviyenin girişi oalrak kullanılamaz. Örneğin, iki Ve kapısının çıkışı bir diğer VE kapısının girişleri olarak kullanılamaz. Çıkışlar istenen formata göre yeniden üretilmelidir, ki bu oldukça maliyetli olur. [8]'de bu problem tartışılmıştır. Üretilen giriş işaretleri -çalışmada PWM işaretleri olarak adlandırılmıştır- çıkışta dizilişlerini kaybederler. Bunun için bir çözüm teklif edilse de, bu çözüm de ekstra girişlere ihtiyaç duyar, bu da yeni katarların üretilmesi gerektiği anlamına gelir. Neticede sunulan çözüm hem tasarım karmaşıklığını arttırmış, hem de hızı önemli miktarda düşürmüş olur.

Literatürdeki bit katarlarının istenen formatta üretilmesine odaklanan yöntemlerin aksine, bu çalışmada teklif edilen BKH için yalnızca katarların taşıdığı değer önemlidir, bu sayede herhangi bir formattaki giriş katarları doğrudan kullanılabilir. Böylece özel katar üreteçlerine olan ihtiyaç ortadan kalkar. Ayrıca, çok seviyeli tasarımlar için ek maliyet de olmaz, bir seviyenin çıkışı diğer seviye için

doğrudan kullanılabilir. [9]'de bu mantığa sahip olan "Alaghi toplayıcısı" adlı bir yüksek doğruluklu toplayıcı teklif edilmiştir. Bu çalışmada teklif edilen senkron toplayıcılardan biri zamanlama üzerinde yapılan değerlendirmeler haricinde bu toplayıcıya oldukça benzerdir. Bu toplayıcının herhangi sayıda giriş katarı için genelleştirilmiş hali de bu çalışmada mevcuttur. Toplayıcılara ek olarak, iki senkron çarpıcı da teklif edilmiştir.

Bahsi geçen bütün tasarımlar doğruluğu geliştirmek amacıyla saat işareti kullanmışlardır, yani bunlar senkron devrelerdir. Bu çalışmada senkronizasyon maliyetinden kurtulmak adına gecikme elemanları ile tasarlanan asenkron toplayıcı ve çarpıcılar da teklif edilmiştir.

Bu tasarımların bir diğer kusuru da ardışık bit katarlarını işleyememektir. Sadece bir seferlik işlemleri yerine getirecekleri farzedilmiştir. Bu kusuru gidermek adına, bu çalışmada birer toplayıcı ve çarpıcı da ardışık bit katarlarını işleme kabiliyetine sahip olarak tasarlanmıştır.

Bahsi geçen bütün kusurlardan ayrı olarak, literatürde zamanlama problemleri hafife almak gibi bir eğilim de mevcuttur. Bu problemler SH ve BKH'ye özgüdür, bunların çözülmemesi teklif edilen çalışmanın uygulanabilirliğini iddia etmeyi güçleştirir. Örneğin [8]'deki tasarımın 1GHz'lik giriş işaretleri ile çalıştığı iddia edilmiştir. 8 bitlik ikili işlemlere eşdeğer işlemlerin yapıldığı varsayıldığında, katarlar için en az 256 farklı değerin bulunması gerekir. Bu yüzden, en kötü senaryoda 1/256 değerini temsil etmek için bir bit katarı $1/(256 \times 10^9)$ saniyelik süreye sahip bir 1 değerli bite sahip olacaktır. Bu da teklif edilen devrenin 0.256 THz işaretli güvenli bir şekilde işleyebilmeleri gibi pek mümkün olmayan bir duruma işaret etmektedir (bkz. Şekil 1.3.a)). Bu yüzden bu çalışma ancak çok daha düşük frekanslarda çalıştırılabilir ki, bu da deneysel sonuçlar bölümünde gösterildiği gibi çarpıcı alan artışına sebep olacaktır. Benzer problemler literatürdeki neredeyse bütün çalışmalarda görülebilir, bit katarlarının frekansları bir katardaki bir bitin süresini devrenin gecikmesinden çok büyük yapacak kadar küçük seçilmediği sürece devre düzgün çalışmayacaktır. Bu sebeple bu çalışmada teklif edilen bütün devreler transistör seviyesinde zamanlama benzetimleri ile tasarlanmıştır.

## 1.3 Genel Bakış

Bu çalışmada BKH ile tasarlanan toplamda üçer adet toplayıcı ve çarpıcı devre teklif edilmiştir. Birer adet toplayıcı ve çarpıcı asenkron devre iken, geri kalanı senkrondur. 4 senkron toplayıcı ve çarpıcı içinde birer toplayıcı ve çarpıcı ardışık giriş bit katarlarını işleme yetisine sahiptir. Teklif edilen bütün tasarımlar AMS 0.35um CMOS teknolojisinde transistör seviyesinde benzetimler yapılarak selefleri ile karşılaştırılmıştır. Ayrıca teklif edilen devreler bir yapay sinir ağı uygulamasında test edilmiştir.

Tezin geri kalanı şu şekilde sıralanmıştır: 2. Bölüm BKH için tanımlar, açıklamalar ve kısaltmalardan oluşmaktadır. 3. ve 4. Bölümde aritmetik işlemler icra eden sırasıyla asenkron ve senkron devreler sunulmuştur. 5. Bölümde teklif edilen devrelerin değerlendirilmesi için deneysel sonuçlar verilmiştir. Son olarak 6. Bölüm bu çalışmadan çıkarılan sonuçları ve çalışmanın hangi yönde ilerleyebileceğine dair tahminleri içermektedir.

## 2. SH VE BKH TEKNİKLERİ İÇİN ÖNCÜLLER

Bu tez içerisinde sıklıkla kullanılmaları sebebiyle **bit genişliği** ve **katar uzunluğu** terimlerinin tanımlarının yapılmasına ihtiyaç duyulmuştur. Bit genişliği, bir katardaki tek bir bitin süresi olarak; katar uzunluğu ise bir katardaki toplam bit sayısı olarak tanımlanmıştır.

1. Bölümde bahsedildiği gibi doğruluk problemi SH'nin en yaygın problemidir ve doğruluğu arttırmak bu çalışmanın birincil amaçlarından biridir. Doğruluğu arttırmak için herhangi bir yöntem teklif etmeden önce SH ile maksimum ne kadar doğruluk elde edilebileceğinin belirlenmesi faydalı olacaktır. Sıradaki teoremde doğruluğu geliştirilmesi için temel bir sınır belirlenmiştir.

**Teorem 1.** *İdeal elemanlarla ideal SH icra eden bir sistem varsayalım. Bu sistemin doğruluğu sadece çıkışın beklenen değeri $z_e$ ve çıkış katar uzunluğu n'e bağlıdır.*

**Kanıt.** *SH'de $z_e$ her bir çıkış bitinin 1 olma olasılığı olarak da tanımlanabilir. Bu yüzden, her bir çıkış biti Bernoulli dağılımına, çıkış katarı ise Binom dağılımına sahiptir. Standart hata (standart sapma) ve bağıl hata sırasıyla $\sqrt{\frac{p\times(1-p)}{n}}$ ve $\sqrt{\frac{(1-p)}{p\times n}}$ olarak hesaplanabilir, ikisi de sadece $p = z_e$ ve n'e bağlıdır.* □

Bu teorem ideal durumu açıkladığından, çıkış katarının dağılımını değiştiren [10] gibi veya korelasyonu azaltmak için kaydırılmış ancak tamamen bağımsız olmayan bit katarlarını kullanan [2] gibi çalışmalar için kullanılamaz. Ancak bu çalışmalar da katarlarının olasılıksal Bernoulli'ye benzer davranışlarından dolayı hataya açıktırlar.

Nispeten basit olan bu teoremden şu sonuçlar çıkarılabilir:

- Sonsuz katar uzunluğuna ihtiyacı olan SH ile hatasız hesaplama imkansızdır.
- Katar uzunluklarını -dolayısıyla hesaplama sürelerini- $X$ kat arttırmak hata oranını sadece $\sqrt{X}$ kat düşürecektir.
- Yüksek doğruluk elde edilmesi için çıkıştaki rasgelelikten feragat edilmelidir.

Bu çalışmada, yukarıdaki sonuçlardan yola çıkılarak "Bit Katarı Hesaplama (BKH) adlı yeni bir hesaplama paradigması sunulmuştur. Bu paradigma rasgele ya da binom dağılımlı giriş/çıkış katarlarını kullanmak zorunda olmadan SH mantığından faydalanır. Stokastik veya deterministik tipte herhangi bir katar kullanılabilir. BKH ile toplama ve çarpma işlemleri icra eden yüksek doğruluklu/hatasız devreler aşağıda verilen kısıtılamalar da göz önüne alınarak tasarlanmıştır. Bit katarlarının değerleri $[0,1]$ aralığında olduğundan toplama işlemi giriş değerlerinin ortalaması alınarak ölçeklenmelidir. Bu da aslında bütün SH veya BKH tasarımlarında toplama yerine ölçekli toplama yapılmasının sebebidir. Bu tez boyunca "ölçekli toplama" terimi yerine doğrudan "toplama" kelimesi kullanılmıştır.

**Lemma 1.** *$0/n$ ve $n/n$ arasında $n+1$ farklı değer alabilen n uzunluklu iki bit katarı varsayılsın. Bu katarların BKH ile hatasız toplanması en az $2\times n$ uzunluklu çıkış katarı gerektirir.*

**Kanıt.** *En kötü senaryolardan biri olan bir girişin $1/n$ diğer girişin ise $0/n$ değeri aldığı durumu düşünelim. Beklenen çıkış değeri $1/(2\times n)$'yi temsil edebilmek için $2\times n$ bit gerekir.* □

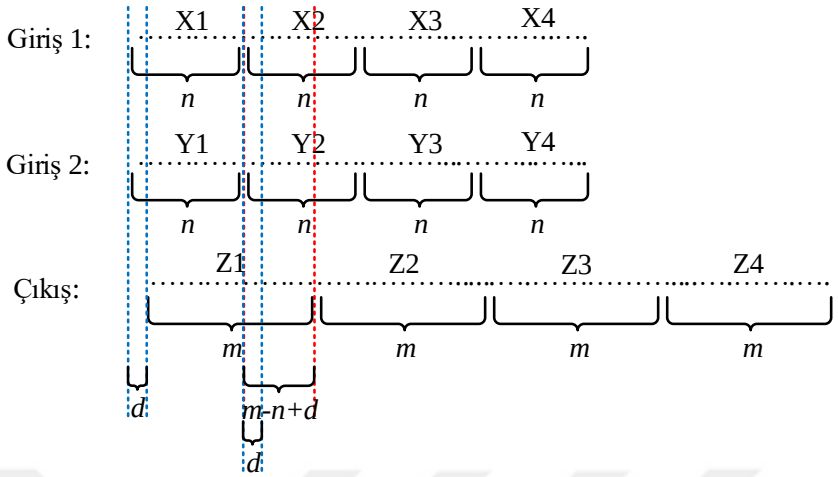
**Lemma 2.** *$0/n$ ve $n/n$ arasında $n+1$ farklı değer alabilen n uzunluklu iki bit katarı varsayılsın. Bu katarların BKH ile hatasız çarpılması en az $n^2$ uzunluklu çıkış katarı gerektirir.*

**Kanıt.** *En kötü senaryo olan iki girişin de $1/n$ değeri aldığı durumu düşünelim. Beklenen çıkış değeri $1/(n^2)$'yi temsil edebilmek için $n^2$ bit gerekir.* □

**Teorem 2.** *$n<m$ olmak üzere, giriş ve çıkış katar uzunlukları sırasıyla n ve m olan BKH icra eden bir sistem varsayalım. Eğer sistemin reaksiyon süresi veya gecikmesi geçmişten bağımsızsa, bu sistem ardışık bit katarlarını doğru bir şekilde işleyemez.*

**Kanıt.** *Bu sistemin d bit kadar gecikmesi olduğu varsayılsın (d ondalıklı olabilir). Bu durumda bit katarlarının girişlere verildiği andan itibaren, ilk çıkış bitinin alınması için sistemin d bitlik bir süre kadar beklemesi gerekir. İki ardışık giriş bit katarı kümesi düşünelim. İlk kümenin girişi tamamlandıktan sonra, sistemin işlemi bitirebilmek için $m-n+d$ ek zamana ihtiyacı olur. Ancak, aynı zamanda d bitlik süre sonunda da ikinci giriş kümesi için sonuç vermeye başlaması da gerekir. Bu durum Şekil 2.1'te*

*de gösterilmiştir. Sonuç olarak, doğru sonuçları elde etmek için* $m-n+d=d$ *ve dolayısıyla* $m=n$ *eşitlikleri sağlanmalıdır.*

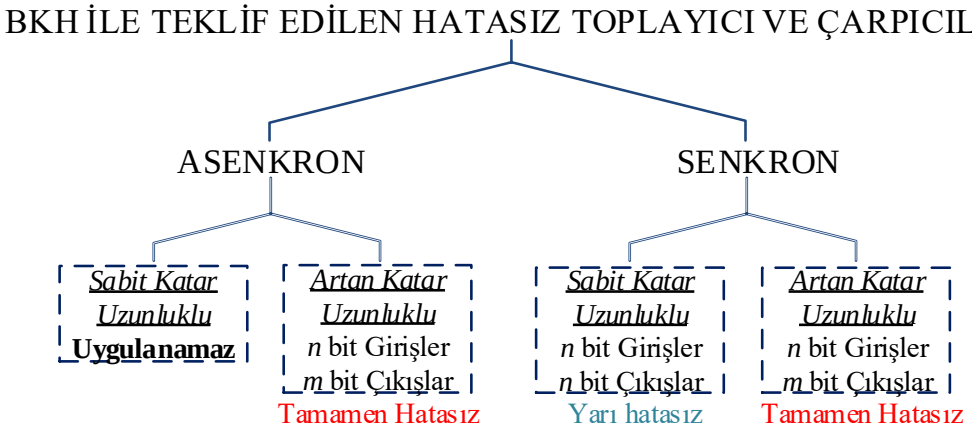



**Şekil 2.1** : Ardışık bit katarlarının işlenmesi.

Teorem 2 ardışık bit katarlarının işlenmesi için iki çözüm ortaya çıkarır. Birincisi, sistemin gecikmesinin sıralı bir biçimde kontrolüdür. Örneğin Şekil 2.1'te ilk, ikinci ve üçüncü giriş katarı kümeleri için sırasıyla $d$, $d+m-n$, ve $d+(2m-n)$ bitlik gecikme olmalıdır. Böyle karmaşık bir sistem doğal olarak BKH'nin alan avantajını ortadan kaldırır. İkinci çözüm ise, giriş ve çıkış bit katarlarının eşit uzunlukta olmasıdır. Bu çözüm sadece alan için değil, aynı zamanda çok seviyeli tasarımlar için de daha iyidir. Bu sebeple bu çalışmada ikinci çözüm tercih edilmiştir.

Aynı uzunlukta katarlar kullanmak hatalı sonuçlara sebep olabilir. Lemma 1 ve 2'de görüldüğü gibi, eğer giriş bit katarları bit uzunluklarının izin verdiği bütün değerleri alabiliyorlarsa, aynı uzunlukta bit katarları kullanılması hatasız toplama ve çarpma icra etmeyi engeller. Örneğin, 16 bitlik giriş ve çıkış katarlarının kullanıldığı durumda çarpma işlemi icra edildiği varsayılsın. İki giriş katarı da 3/16 iken, doğru sonuç $9/256 = 0.5625/16$ olmalıdır. Ancak çıkışta görülebilecek en yakın sonuçlar 0/16 veya 1/16'dır. Diğer bir deyişle çıkışta hata görülmesi mecburidir. Hatanın minimize edilmesi adına çıkış değeri en yakın tam sayıya yuvarlanabilir. BU örnekte yuvarlak sonuç 1/16 olacaktır.

Bu doğruluk problemi göz önüne alınarak bu çalışmada teklif edilen asenkron ve senkron devreler ardışık katar işleyebilen "Yarı-Hatasız" ve daima doğru sonuç veren "Tamamen Hatasız" olarak sınıflandırılmıştır. Tamamen hatasız devrelerin yukarıda




**Şekil 2.2** : Teklif edilen toplayıcı ve çarpıcı tasarımlarının özeti.

bahsi geçen çözümü uygulamadığı, yani ardışık işlem için uygun olmadığı dikkate alınmalıdır.

# 3. ASENKRON TOPLAYICI VE ÇARPICILAR

İlk olarak, Şekil 2.2'te görüldüğü gibi sabit katar uzunluklu asenkron devrelerin bu çalışmada yer almamasının sebebi açıklanacaktır.

**Lemma 3.** *BKH icra eden, 0'dan 1'e alınabilecek bütün değerleri alabilen birden çok girişli bir sistem varsayılsın. hatasız sonuç üretmek için çıkış katarının uzunluğunun giriş katarlarınınkinden büyük olması gerekiyorsa, giriş katarınındaki bir bitlik değişikliğin çıkış katarındaki birden çok değişikliğe sebep olacağı en az bir durum olacaktır.*

**Kanıt.** *Girişteki bir bitlik değişikliğin çıkışta da en fazla bir bitlik değişikliğe sebep olacağı varsayılsın. Bu durumda hatasız sonuç üretmek için giriş ve çıkış katarları eşit uzunlukta olmalıdır. Bu da bir çelişkili bir durum ortaya çıkarır.*

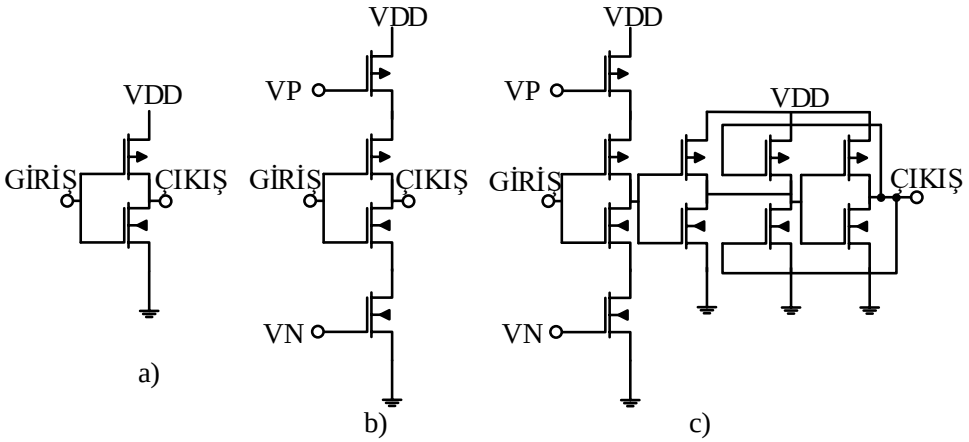
Lemma 3'in doğrudan sonucu, toplama ve çarpma işlemlerinde eşit uzunlukta bit sayısı kullanabilmek amacıyla giriş bit bilgilerinin gelecekte kullanılması adına saklanması gerektiğidir. Bu ardışıl işlem asenkron devreler için çok maliyetli olduğundan, bu çalışmada sabit katar uzunluklu asenkron devreler tasarlanmamıştır.

Ayrıca, bu çalışmadaki tasarımlar için aşağıdaki kısıtlama da göz önüne alınmıştır.

**Lemma 4.** *BKH icra eden tamamen hatasız asenkron bir sistemde giriş ve çıkış katar uzunluklarının sırasıyla n ve m olduğu varsayılsın ($n < m$). Sistem, içerisindeki devre elemanlarında en az $m-n$ giriş/çıkış düğümü barındırmalıdır.*

**Kanıt.** *Çıkışın n. bitine sıra geldiğinde, geriye kalan $m-n$ bit sistemin ihtiyaç duyacağı $m-n$ düğümde saklanmalıdır. Asenkron sistem geriye kalan bilginin $m-n$ den az düğümde saklanıp yeniden işlenmesine olanak tanımaz.*

Lemma 1 ve 2'te verilen aritmetik işlemler ve bu işlemlerin katar uzunluklarına dair kısıtlamalar göz önüne alındığında, tamamen hatasız toplayıcı ve çarpıcı devreler için sırasıyla $n$ ve $n^2-n$ düğüme ihtiyaç duyduğu sonucu çıkarılabilir. Lojik kapılar devre elemanları olarak kullanıldığından ve eviriciler bunlar arasında en çok alan




**Şekil 3.1** : Gecikme elemanı olarak evirici: a) geleneksel evirici, b) NP-gerilim kontrollü evirici, c) Schmitt tetikleyici ile kaskat bağlanmış versiyonu.

dostu olanlar olduklarından, Lemma 4'te bahsedildiği gibi $m - n$ bitin saklanması için gecikme elemanı olarak eviriciler kullanılmıştır. Çift sayıdaki eviricilerin kaskat kullanılması ile istenen gecikme miktarına ulaşılabilir.

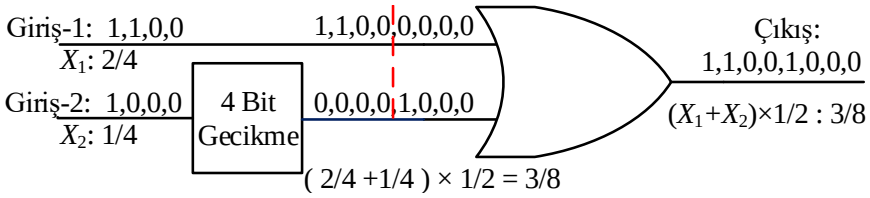
Uygun evirici tasarımını seçebilmek amacıyla 3 kriter göz önüne alınmıştır:

- Devre alanı BKH ile uyumlu olacak kadar küçük olmalı,
- Yükselme ve düşme gecikmeleri işaretin bütünlüğünün korunması açısından düşük olmalı,
- Gecikme değerlerindeki sapmaları giderebilmek için gecikme miktarı kontrol edilebilmeli.

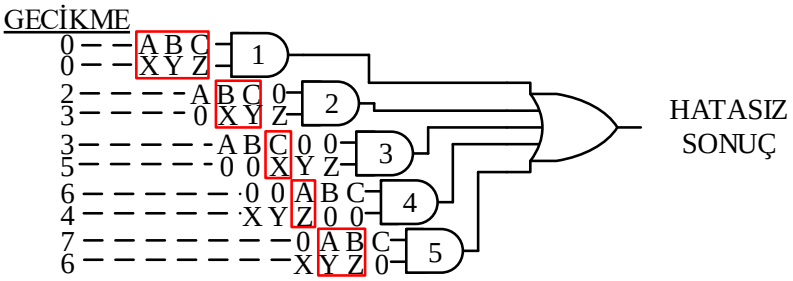
Farklı seçeneklerin değerlendirilmesi sonucunda, Şekil 3.1'te gösterilen 3 farklı evirici tasarımı öne çıkmıştır. Gelenseksel eviricinin gecikme kontrolü Şekil 3.1.a)'da gösterildiği gibi VDD ölçeklendirmesi ile sağlanabilir. Daha iyi bir kontrol için, Şekil 3.1.b)'deki gibi VN ve VP analog gerilim girişleri kullanılabilir. EK olarak, işaret bütünlüğünün elde edilmesi amacıyla NP kontrollü evirici Şekil 3.1.c)'deki gibi Schmitt tetikleme devresine kaskat bağlanabilir. Bu 3 seçenek içerisinden basitlik ve alan tasarrufu için ik geleneksel evirici tercih edilmiştir.

### 3.1 Artan Katar Uzunlukları ile Tamamen Hatasız Toplama

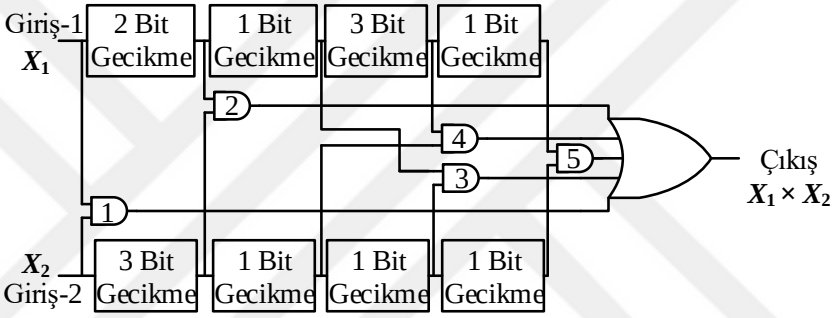
Teklif edilen toplayıcı Şekil 3.2'te görüldüğü gibi birer gecikme bloğu ve VEYA kapısından oluşmaktadır. Gecikme bloğu girişlerden birinin giriş katarının kapladığı




**Şekil 3.2** : Teklif edilen asenkron toplayıcı; $n = 4$.




**Şekil 3.3** : 3 bitlik girişler için asenkron çarpma işleminin izahı.




**Şekil 3.4** : 3 bitlik girişler için asenkron çarpıcı devresi.

toplam süre ((giriş katar uzunluğu $n$)$\times$(bit genişliği) şeklinde hesaplanabilir) kadar geciktirilmesi için kullanılmıştır. Bu süreyi ve Lemma 4'i sağlamak için, $n$ çift veya tek sayı olduğunda sırasıyla en az $n$ veya $n+1$ eviricinin kullanılması gerekir. Sonuç olarak devrenin alan karmaşıklığı $O(n)$'dir.

## 3.2 Artan Katar Uzunlukları ile Tamamen Hatasız Çarpma

Hatasız çarpma işlemi ancak giriş katarlarının birindeki her bir bitin diğerindeki her bir bitle tek tek çarpılması ile garanti edilebilir. Bu yüzden $n$ bitlik giriş katarları için toplam $n^2$ işleme ihtiyaç vardır. Bu şart, 1 tanesi gecikmenin olmadığı, $n$ tanesi bir katarın diğerinden daha fazla gecikmeye sahip olduğu, $n$ tanesi ise tersi durumun geçerli olduğu toplam $2n-1$ farklı durum için giriş katarlarına gecikme uygulanarak sağlanabilir. VE kapıları ile çarpma işlemi sağlandıktan sonra, bir VEYA kapısı sonuçları birleştirmek için kullanılabilir. Bu yöntem $n = 3$ için Şekil 3.3'te gösterilmiştir. Toplam 5 farklı durum 5 adet VE kapısına karşılık gelmektedir. Şekil 3.4'te ise aynı yöntemin devre yapısı gösterilmiştir.

$n$ bitlik girişler için, teklif edilen devre 4 adımda tasarlanabilir:

1. Girişler herhangi bir yapmadan VE kapısından geçirilir (Şekil 3.3'teki 1 numaralı VE kapısına karşılık gelir.).

2. $i = 1,2,...,n-1$ için Giriş-1'in son $n-i$ biti ile Giriş-2'nin ilk $n-i$ biti VE kapısından geçirilir (Şekil 3.3'teki 2 ve 3 numaralı VE kapılarına karşılık gelir). Bu işlemler için toplamda Giriş-1 için $n-1$, Giriş-2 için $n-1$ adet gecikme bloğu kullanılmıştır (Şekil 3.3'in üst ve alt kısmındaki 2'şer gecikme bloğuna karşılık gelir).

3. $i = 1,2,...,n-1$ için Giriş-1'in ilk $n-i$ biti ile Giriş-2'nin son $n-i$ biti VE kapısından geçirilir (Şekil 3.3'teki 4 ve 5 numaralı VE kapılarına karşılık gelir). Bu işlemler için toplamda Giriş-1 için $n-1$, Giriş-2 için $n-1$ adet gecikme bloğu kullanılmıştır (Şekil 3.3'in üst ve alt kısmındaki 2'şer gecikme bloğuna karşılık gelir).

4. $2n-1$ adet VE kapısının tamamının çıkışları $2n-1$ girişli VEYA kapısından geçirilir. Bu VEYA kapısının çıkışı hatasız sonucu verir.

$i.$ ve $(i-1).$ VE kapıları arasındaki gecikme farkları da aşağıdaki gibi genelleştirilebilir:

$$\text{Giriş-1 için } \begin{cases} 0 & i = 1 \\ n-(i-1) & i = 2,3,\ldots,n \\ n & i = n+1 \\ i-(n+1) & i = n+2,n+3,\ldots,2n-1 \end{cases}$$

$$\text{Giriş-2 için } \begin{cases} 0 & i = 1 \\ n-i & i = 2,3,\ldots,n \\ -(n-2) & i = n+1 \\ i-n & i = n+2,n+3,\ldots,2n-1 \end{cases}$$

Giriş-2 için $i = n+1$ durumunda negatif değer görülmesinin, $i = n$ durumundan daha az gecikmeye ihtiyaç duyulduğu anlamına geldiği dikkate alınmalıdır (bkz. Şekil 3.3 ve 3.4).

Sonuç olarak, gecikme blokları Giriş-1 ve Giriş-2 için toplamda sırasıyla $n^2-(n-1)$ ve $n^2-(n)$ bit gecikmeye yol açarlar.Bu yüzden bu gecikme bloklarının gerçeklenmesi için en az $\approx 2n^2$ eviriciye ihtiyaç vardır. Bu durumda alan karmaşıklığı $O(n^2)$ olur.

# 4. SENKRON TOPLAYICI VE ÇARPICILAR

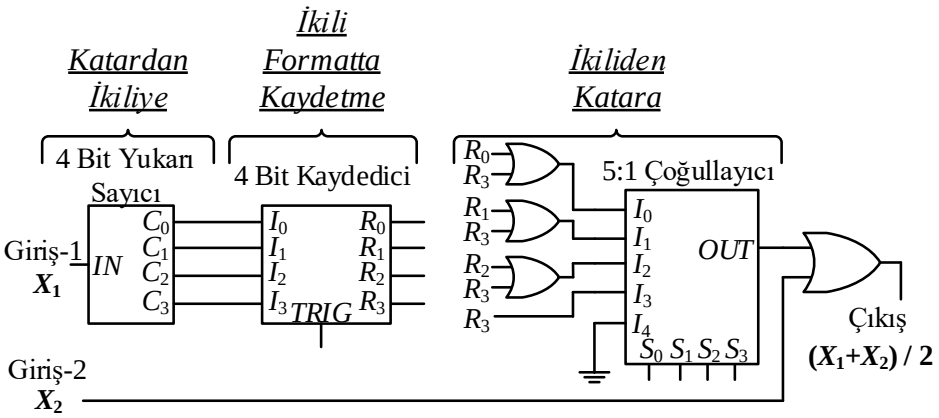
Teklif edilen asenkron devrelerin gecikme kontrollü özellikleriyle tasarımları kolaydır. Ancak alan masrafları giriş bit katar uzunluğu $n$'in artmasıyla üssel olarak artar. Ayrıca, ardışık giriş katarlarını işlemeye de uygun değillerdir. Bu problemleri çözmek amacıyla senkron tasarımlar bu çalışmanın ilgisi alanındadır. Yardımcı işaretlerin varlığı katarlarda taşınan bilgilerin ikili rakamlarla tutulması ve işlenmesine izin verir. Tasarlanan senkron devreler hem katarların hem de ikili rakamların işlendiği devrelerdir.

Teklif edilen senkron tasarımlar artan uzunluklu ve sabit uzunluklu bit katarları içerenler olmak üzere iki gruba ayrılır. Artan katar uzunluklu devreler Bölüm 3'teki asenkron devreler gibi tamamen hatasız iken, sabit katar uzunluklular ardışık bit katarlarını işleyebilmek adına düşük de olsa bir miktar hataya sahiptirler. Sonuç olarak aşağıdaki 4 alt bölümde teklif edilen 2'şer adet toplayıcı ve çarpıcı devre ayrıntılı biçimde açıklanmıştır.

## 4.1 Artan Katar Uzunluklu Tamamen Hatasız Toplama

Asenkron toplayıcı için kullanılan yaklaşım bu tasarım için de temel alınmıştır: giriş katarlarının biri diğerinin bütün bitleri işlenene kadar bekletilir. Şekil 3.2'teki asenkron gecikme blokları yerine, özellikle büyük katar uzunlukları için alan konusunda çok daha verimli, giriş bilgisini ikili formatta saklayan senkron bloklar kullanılmıştır.

Şekil 4.1, giriş bit katarı uzunluğu $n = 8$ için teklif edilen toplayıcıyı göstermektedir. Burada $X_1$ ve $X_2$, $0/8$ ve $8/8$ arasında değişen giriş değerlerini temsil etmektedir. Lemma 1'te belirtildiği gibi, hatasız toplama için çıkış katar uzunluğunun 16 olması gerekir. Teklif edilen toplayıcı öncelikle $X_1$'i bir sayıcı aracılığıyla ikili formata çevirir. Sayma işleminin sonunda, kaydedici sayıcının çıkışındaki bilgiyi kaydeder. Ardından ikili sayıdan bit katarına çevirme işlemi çoğullayıcı ile yapılır. Son olarak, bir VEYA kapısı ile toplama işlemi icra edilmiş olur.




**Şekil 4.1** : 8 bitlik girişler için teklif edilen senkron tamamen hatasız toplayıcı.

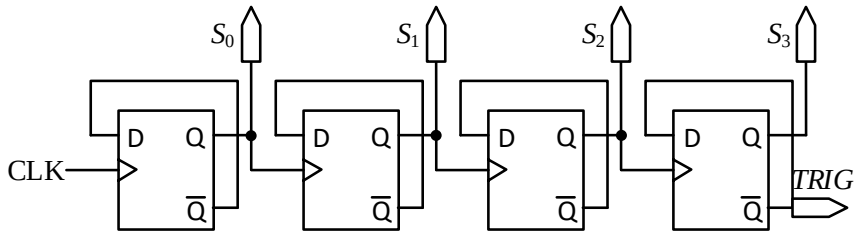
**Çizelge 4.1** : Şekil 4.1'teki 5:1 çoğullayıcının seçim girişleri ile çıkışı arasındaki ilişki.

| SEÇİM GİRİŞLERİ | | | | ÇIKIŞ | |
|---|---|---|---|---|---|
| $S_3$ | $S_2$ | $S_1$ | $S_0$ | Karşılık | Süre |
| 0 | 0 | 0 | 0 | $I_0$ | 1 bit |
| 0 | 0 | 0 | 1 | $I_3$ | 1 bit |
| 0 | 0 | 1 | X | $I_1$ | 2 bit |
| 0 | 1 | X | X | $I_2$ | 4 bit |
| 1 | X | X | X | $I_4$ | 8 bit |

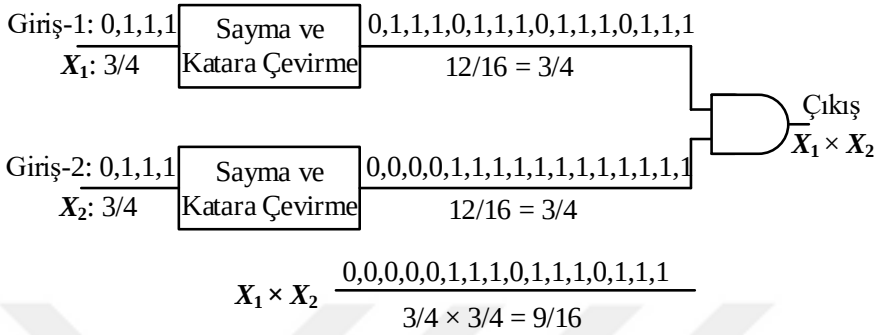
8 bitlik katar 3 bitlik ikili çözünürlüğe tekabül etse de, hem sayıcı hem de kaydedici $0/8$ ile $8/8$ arasındaki 9 değerin tamamını temsil edebilmek amacıyla 4 bitlik seçilmiştir. Bu durum ayrıca çoğullayıcıdan önce kullanılan VEYA kapılarının da sebebidir. Kaydediciden gelen en büyük ikili değer çoğullayıcının çıkışındaki katarın bütün bitlerinde 1 üretilmesi gereken 1000 ($R_3 = 1$) değeridir. Bu amaçla, $R_3$ diğer bütün $R$'ler ile tek tek VEYA kapısından geçirilmiştir.

Çoğullayıcıda kaydediciden gelen 4 girişle beraber, Giriş-2'den gelen giriş katarını işlemek amacıyla $n$ bitlik süre boyunca çıkışa lojik 0 verilmesi için kullanılan bir giriş daha bulunur. Çizelge 4.1 çoğullayıcının veri girişleri, seçim girişleri ve çıkışı arasındaki ilişkiyi göstermektedir. Bütün seçim girişleri aslen %50 görev çevrimli saat işaretleridir. Şekil 4.2'te gösterilen devredeki gibi tek bir CLK işaretinden üretilebilirler. Kaydedicinin $TRIG$ girişi ise $\overline{S_3}$ olarak seçilmelidir.

Giriş katarları $n$ bite sahip olduklarında, sayıcı ve kaydedici $\log_2 n + 1$ bit olmalı, çoğullayıcı ise $\log_2 n + 2$ veri girişine ve $\log_2 n + 1$ seçim girişine sahip olmalıdır. Bütün yardımcı işaretler de $\log_2 n + 1$ ardışık flip-flop içeren frekans bölücü devresinden üretilebilir. Neticede alan karmaşıklığı $O(\log n)$ olur.




**Şekil 4.2** : Şekil 4.1'teki yardımcı işaretlerin üretilmesi için kullanılan devre.



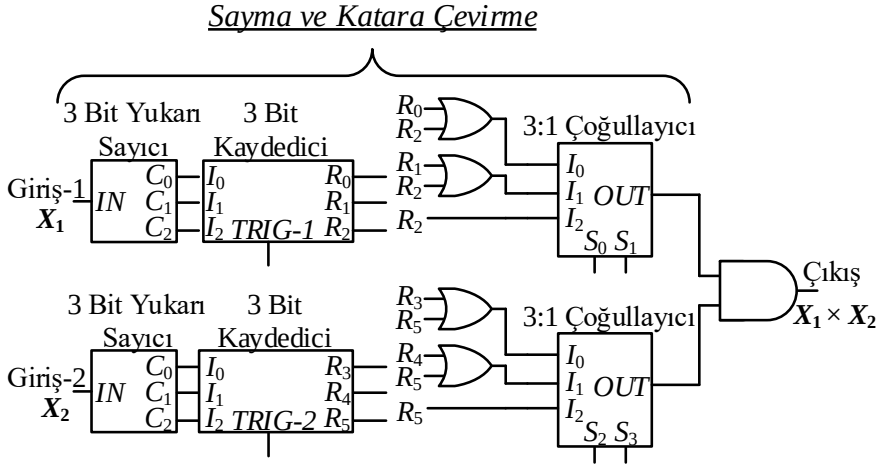

**Şekil 4.3** : 4 bitlik girişler için teklif edilen çarpım şeması.

## 4.2 Artan Katar Uzunluklu Tamamen Hatasız Çarpma

Uzunluğu $n$ olan giriş bit katarları varsayılsın. Daha önce bahsi geçtiği gibi, hatasız çarpma $n^2$ adet bit bit işlem gerektirir ki, bu Lemma 2'e de uygundur. Bu şart katarlardan birini $n$, diğerinin de tek tek bütün bitlerini $n$ kez tekrarlayarak sağlanabilir. $n = 4$ için örnek Şekil 4.3'te gösterilmiştir. Giriş katarlarındaki 0 ve 1'lerin sıralamalarının tekrarlanan katarlara yansıtılmadığı dikkate alınmalıdır; sayım işleminden sonra elde kalanlar sadece giriş değerleri $X_1$ ve $X_2$'dir, sıralamalar kaydedilmemiştir. Şekildeki örnekte $X_1 = X_2$ olduğundan giriş katarlarının ikisine de (0,1,1,1)'miş gibi davranılmıştır.

$n = 4$ için teklif edilen çarpıcı devresi Şekil 4.4'te verilmiştir. Sayma ve çevirme devresi Şekil 4.1'teki ile neredeyse aynıdır. Tek fark çoğullayıcılardaki giriş sayısının bir düşük olmasıdır, çünkü toplayıcıda olduğu gibi girişlerden birinin bekletilmesi gerekmemektedir. Üstteki çoğullayıcının seçim girişleri ($S_0$, $S_1$) alttakinin seçim girişlerinden ($S_2$, $S_3$) 4 kat daha hızlıdır. Ek olarak *TRIG-1* girişi $S_3$'ün iki kat yavaşlatılmış halinin tersidir. Böylece bütün yardımcı işaretler 5 ardışık flip-flop içeren bir frekans bölücü devre ile yalnızca bir saat işareti kullanılarak üretilebilir.

Teklif edilen çarpıcının ölçeklenebilirliği analiz edildiğinde, sayıcı ve kaydedicilerin $\log_2 n + 1$ bit, çoğullayıcının da $\log_2 n + 1$ girişli olması gerektiği görülür. Ayrıca




**Şekil 4.4** : 4 bit girişler için teklif edilen tamamen hatasız senkron çarpıcı.

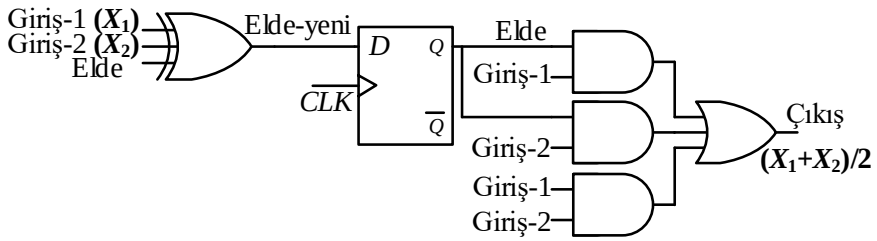
**Çizelge 4.2** : Teklif edilen toplayıcının geçiş tablosu

| Elde | Giriş-1 | Giriş-2 | Çıkış | Elde-yeni |
|---|---|---|---|---|
| X | 0 | 0 | 0 | Elde |
| X | 0 | 1 | Elde | $\overline{\text{Elde}}$ |
| X | 1 | 0 | Elde | $\overline{\text{Elde}}$ |
| X | 1 | 1 | 1 | Elde |

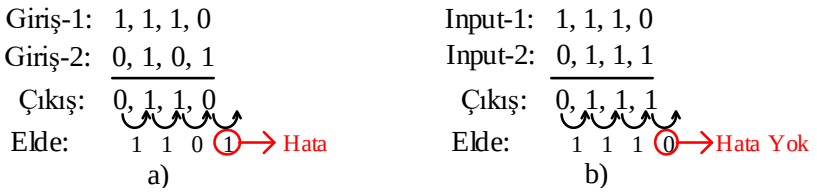
$\log_2 n+2$ ardışık flip-flop da ihtiyaç duyulan bütün yardımcı işaretlerin üretilmesi için kullanılır. Böylece alan karmaşıklığı $O(\log n)$ olur.

### 4.3 Sabit Katar Uzunluklu Yarı Hatasız Toplama

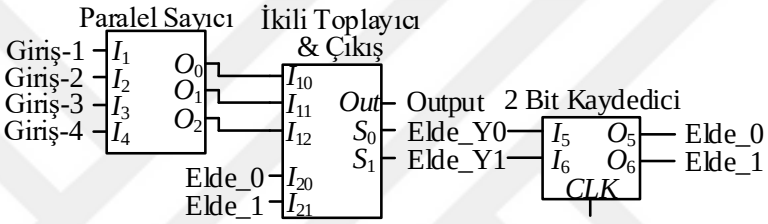
Esasen BKH'de toplama işlemi giriş değerleri $X_1$ ve $X_2$'nin ortalamasını almaktır. Sabit katar uzunluklu tasarımlarda bu girişlerin bit bit ortalaması alınarak sağlanabilir. Ancak biri 1 diğeri 0 olan iki bitin ortalaması 0.5 olacağından ve bu da tek bir bit ile temsil edilemeyeceğinden, arta kalan bilginin saklanması için "elde"ye ihtiyaç duyulur. Çizelge 4.2 böyle bir çözümün doğruluk tablosunu göstermektedir. Bu çizelgeye uyumlu herhangi bir devre beklenen toplayıcı davranışına sahip olacaktır. Şekil 4.5 teklif edilen toplayıcı devresini göstermektedir. Devre şu şekilde çalışır: eğer giriş katarlarından ikisi de çift veya tek sayıda 1 içeriyorsa, sonuç tamamen doğrudur; aksi halde çıkış doğru sonucun $0.5/n$'lik bir hata ile yuvarlanmış versiyonudur ($n$: giriş katar uzunluğu). Şekil 4.6 teklif edilen toplama işleminin hatalı ve hatasız sonuç verdiği birer örnek içermektedir. Devrenin alanı giriş katar uzunluklarına göre değişmediğinden, alan karmaşıklığı $O(1)$'dir.




**Şekil 4.5** : Teklif edilen iki girişli yarı-hatasız toplama devresi.



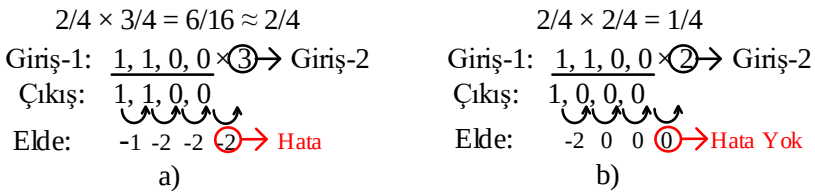

**Şekil 4.6** : Teklif edilen toplama işlemi örnekleri: a) $X_1 = 3/4$ ve $X_2 = 2/4$, b) $X_1 = 3/4$ ve $X_2 = 3/4$.




**Şekil 4.7** : 4 girişli durum için teklif edilen yarı hatasız senkron toplayıcı.

Teklif edilen Şekil 4.5'teki toplama devresi [9]'deki ölçekli toplayıcı ile oldukça benzer başarıma sahiptir. Ancak, bu çalışmadaki bakış açısı ile toplayıcı $i$ adet girişli toplayıcı olarak genelleştirilebilir. Yüksek doğruluklu çok-girişli ölçekli toplama işlemleri icra eden bir yöntem sunan başka bir çalışma da mevcuttur [11]. Ancak, bu çalışma da her durum için doğru sonuca en yakın sonucu vermez, çünkü çıkışta her bir 1 üretimi için modulo-$n$ sayıcıya doyurmaya ihtiyaç duyar. Bu yönteme bir örnek olarak; 4-girişli, girişlerden üçünün 0/4 birinin de 3/4 olduğu bir ölçekli toplayıcı 0/4 sonucunu verir. Ancak, doğruya en yakın sonuç Şekil 4.7'te teklif edilen devrede elde edileceği gibi 1/4'tür.

Bu tasarımda, öncelikle giriş bitleri paralel olarak sayılıp sonuç "elde"ye eklenir. "elde"ye olası negatif elde değerlerinden kaçınmak için $i/2$ başlangıç değeri verilir. Eldenin $i$'den büyük olduğu durumda, çıkışa 1 verilir ve eldeden $i$ çıkarılır. Aksi halde çıkışa 0 verilir. Şekil 4.7 $i = 4$ durumunda devrenin alacağı yapıyı göstermektedir. Buradaki "Paralel Toplayıcı" giriş katarlarındaki 1'leri sayar. "İkili Toplayıcı & Çıkış" bloğu ise mevcut elde değerini sayıcının çıkışına ekler, ardından çıkışı belirlerek elde değerini bahsi geçen işlem basamaklarına göre günceller.




**Şekil 4.8** : Teklif edilen çarpma yaklaşımı için örnekler: a) $X_1 = 2/4$ ve $X_2 = 3/4$, b) $X_1 = 2/4$ ve $X_2 = 2/4$.

## 4.4 Sabit Katar Uzunluklu Yarı Hatasız Çarpma

Çarpma işleminin esasen toplama işleminin tekrarlanması olduğu göz önüne alınırsa, toplama işlemindeki yaklaşım çarpma için de kullanılabilir. $n$ katar uzunluğu olmak üzere, giriş bit katarlarının $X_1 = a/n$ ve $X_2 = b/n$ değerlerini temsil ettiği varsayılsın. İlk katarın $b$ kopyası veya ikincinin $a$ kopyası toplanırsa, elde aracılığıyla bit bit ortalama alınarak çarpma işlemi gerçekleştirilebilir. Toplama işleminde en yakın tamsayıya yuvarlama işleminin daima pozitif elde değerleriyle sağlanmasına karşın, en iyi performansı veren çarpma işleminde elde değerleri $-0.5n$ ve $+0.5n$ aralığında değişebilir. Böylece, mutlak hata $0.5/n$ değeri ile sınırlandırılır. Sadece pozitif elde bitlerinin kullanılmasında üst sınırın $1/n$ olacağı dikkate alınmalıdır.

Şekil 4.8 teklif edilen çarpma şemasını 4 bitlik girişler için açıklamaktadır. Örneğin Şekil 4.8.a)'da ilk işlem 3 adet 1'i toplayıp 3 sonucunu almakta, ardından $3/4$ değeri en yakın tamsayı olan 1'e yuvarlanıp çıkışa verilmekte, $3-4=-1$ değerli elde de 3 adet 1 içeren sonraki bit işlemine iletilmektedir. Buradaki toplama işlemi ile de $2/4$ olan yeni elde değeri de 1'e yuvarlanarak çıkışa verilmekte (0'a da eşit uzaklıkta olduğu için yuvarlanabilirdi.) ve elde $2-4=-2$ olarak güncellenmektedir. En son elde değeri olan -2 aynı zamanda yapılan işlemin hatasını da göstermektedir.

Şekil 4.8'teki akışın doğrudan uygulanması, öncelikle Giriş-2'nin işlenmesi ve onun değerine bağlı olarak Giriş-1'in işlenmesini gerektirmektedir. Bunun yerine girişlerin birbirinden bağımsız olarak ayrı ayrı işlenmesi tercih edilmiştir. Böylece girişler aynı anda birbirinden bağımsız devrelerle yeniden üretilerek işlenebilir. Şekil 4.8'teki genel akış daha hızlı, daha sade ve küçük bir çarpıcı devresi ile sağlanmıştır. Aslında giriş işaretlerinin daha iyi doğruluğun elde edilmesi için yeniden üretilmesi [12] ve [7]'te de denenmiştir. Çıkışta %100 doğruluğun elde edilmesi için giriş katarları manipüle edilmiştir. Ancak, bu çalışmalar daha uzun çıkış katarları üretir ki, bu durumun ciddi problemlere yol açtığından 2. Bölüm'de bahsedilmişti.

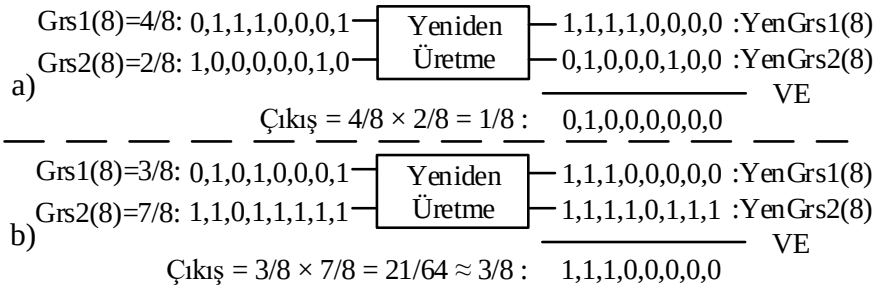
**Algoritma 1** Sabit Katar Uzunluklu Çarpımda En İyi Hata Performansı için Girişlerin Yeniden Üretilmesi

```
1: procedure YENGRS(Grs1(n),Grs2(n))=(YenGrs1(n),YenGrs2(n))
2:     TopGrs1 ← toplam(Grs1)
3:     TopGrs2 ← toplam(Grs2)
4:     Elde ← 0
5:     for i ← 1,n do
6:         if i ≤ TopGrs1 then
7:             YenGrs1(i) ← 1
8:         else
9:             YenGrs1(i) ← 0
10:        end if
11:        Elde ← Elde + TopGrs2
12:        if Elde ≥ n/2 then
13:            YenGrs2(i) ← 1
14:            Elde ← Elde − n
15:        else
16:            YenGrs2(i) ← 0
17:        end if
18:    end for
19: end procedure
```

Teklif edilen çarpma taslağında, giriş katarlarından biri önce 1'leri sonra 0'ları verecek şekilde, diğeri de çarpan olarak kullanılabilcek şekilde yeniden üretilmektedir. Her bit işleminde, çarpan eldeye eklenir. Algoritma 1 giriş katarları $Grs1(n)$ ve $Grs2(n)$'in sırasıyla $YenGrs1(n)$ ve $YenGrs2(n)$ olarak yeniden üretilmesindeki adımları göstermektedir. $YenGrs1(n)$'in üretimi oldukça basittir; önce 1'ler, sonra 0'lar. Ancak $YenGrs2(n)$ daha karmaşık bir biçimde üretilir. Her iterasyonda, $Grs2(n)$'deki toplam bit sayısı $Elde$'ye eklenir. Yeni $Elde$ $n/2$'den küçük değilse, $YenGrs2(i)$ 1 olur ve $Elde$'den $n$ çıkarılır. Aksi durumda $YenGrs2(i)$ 0 olur. İlk iterasyonda $Elde$'nin 0 olduğu dikkate alınmalıdır.

Şekil 4.9 giriş katarlarının yeniden üretilmesine dair iki örnek içerir. Çarpma işlemi yeniden üretilen katarlarınn VE kapsından geçirilmesi ile tamamlanır. Şekil 4.9.a)'da beklenen çıkış değeri olan $1/8$ 8 bit ile temsil edilebildiğinden, hatasız sonuç elde edilebilmektedir. Şekil 4.9.b)'de ise beklenen çıkış değeri $21/64$ 8 bitle hatasız olarak temsil edilemeyeceğinden, çıkışta en yakın temsil edilebilir değer olan $3/8$ görülür.

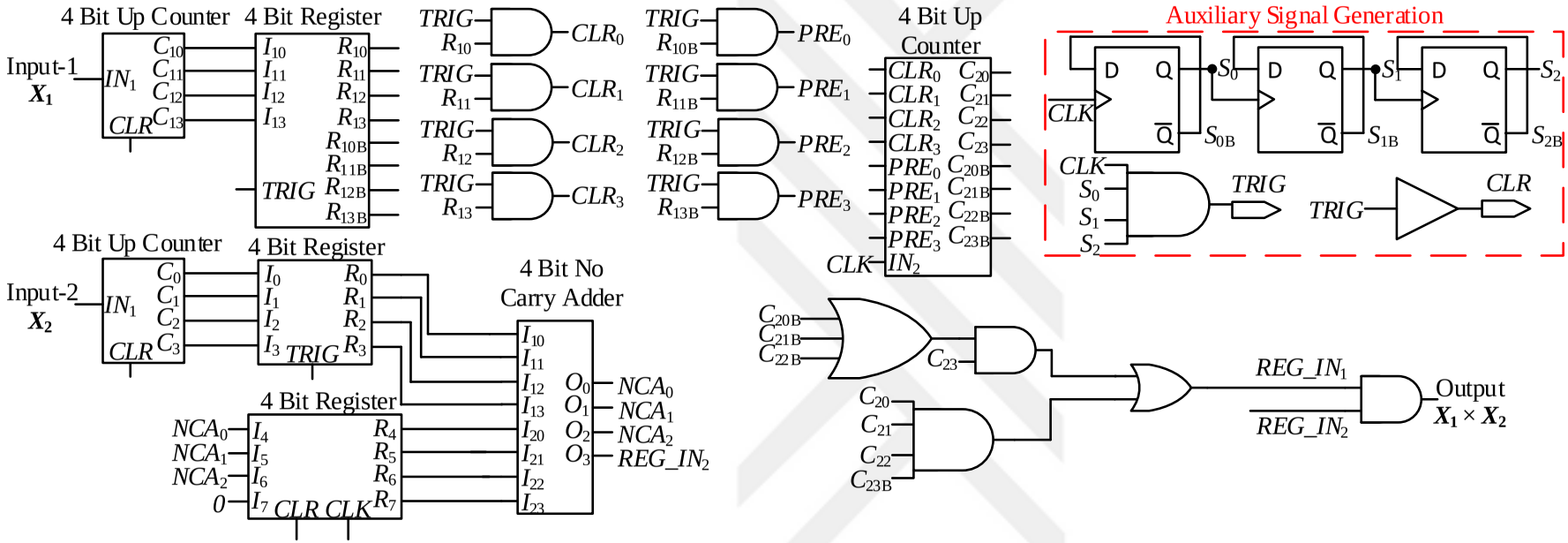
Şekil 4.10 Algoritma 1'i gerçekleyen bir devreyi içermektedir. Devre birbiri ile tamamen aynı olan 3 adet 4-bitlik yukarı sayıcıya sahiptir. Bu sayıcıların her birinde 1 giriş $IN_i$, 4 çıkış $C_{ij}$ ve bunların tersi $C_{ijB}$, 4'er adet silme $CLR_i$ ve yükleme




**Şekil 4.9** : Algoritma 1 aracılığıyla girişlerin yeniden üretilmesi ile gerçeklenen çarpma işlemi a) hatasız işlem, and b) hatalı işlem.

girişleri $PRE_i$ bulunur. Sadelik açısından kullanılmayan giriş ve çıkışlar devrede gösterilmemiştir. Sayıcıların birbirinden farklı görünmesinin sebebi de budur. Benzer şekilde, birbiriyle aynı 3 kaydedicinin de 4 girişi $I_{ij}$, 4 çıkışı $R_{ij}$ ve bunların tersi $R_{ijB}$, silme *CLR* ve saat girişleri *CLK* veya *TRIG*. Kullanılmayan giriş/çıkışlar kaydedicilerde de gösterilmemiştir.

Girişler $IN_1$ ve $IN_2$ öncelikle yukarı sayıcılarda sayılır. $IN_1$ 4 bit kaydedicide saklanır, ardından *TRIG*, $R_{1j}$, ve $R_{1jB}$ ile üretilen $CLR_i$ ve $PRE_i$ girişleri ile ters olarak sonraki 4 bit sayıcıya yüklenir. $CLR_i$ ve $PRE_i$ işaretleri, sayıcıda karşılık gelen D-FF'ların *CLEAR* ve *PRESET* girişlerine, ilk sayıcıdaki sayma işlemi tamamlandıktan sonra kaydedicide saklanan bilginin tersinin sonraki sayıcıya yüklenmesi için bağlanmıştır. Bu da ilgili sayıcının sayma işlemine "0000" yerine kaydedilen bilginin tersinden başlaması anlamına gelir.




**Şekil 4.10** : 8 bitlik girişler için teklif edilen senkron yarı-hatasız çarpma devresi.

Daha sonra sayıcının en anlamlı biti $C_{23}$ ve diğer bitlerin tersleri $C_{2iB}$ ($i = 0,1,2$ için) $YEN\_GRS_1$'in ne olacağını belirler. Eğer sayıcının çıkışı 1001'den küçükse, $YEN\_GRS_1$ 1 olur; aksi halde 0 olur. Örneğin, $X_1 = 5/8$ durumunda, kaydedicide saklanan bilgi ve sayıcıya yüklenen bilgi sırasıyla 0101 ve 1010 olur. Bu durumda sayıcı 1010 değerinden başlayarak yukarı sayar ve 5 saat döngüsünden sonra 0000'a varır, diğer bir deyişle $YEN\_GRS_1$'in ilk 5 biti 1, geri kalanı da 0 olur. Bu sayede, maliyetli bir karşılaştırıcı yerine bir yukarı sayıcı ve ufak bir lojik devre ile $REG\_IN_1$ işareti üretilmiş olur.

4 bitlik kaydedicide kaydedilen bilgi, çıkışındaki en anlamlı biti (diğer bir deyişle son eldesi) çıkarılmış bir ikili toplama bloğu olan "No Carry Adder" aracılığıyla, her saat darbesinde bu toplayıcının bir önceki sonucu ile toplanır. Bu toplama işlemi Algoritma 1'in 11. satırındaki $Elde \leftarrow Elde + TopGrs2$ işleme karşılık gelir. Diğer bir deyişle, şekilde en altta görülen kaydedicinin girişi $Elde$'ye tekabül eder. $Elde$, normal şartlarda $[-n/2, n/2]$ arasında değerler almaktadır. Negatif değerlerden kurtulmak adına, el alttaki kaydedicinin çıkışı 00...0 yerine 01..0'den başlatılarak $Elde$ 0 yerine $n/2$ değerinden başlatılmış olur, böylece $Elde$'nin değişim aralığı $[0,n]$ olur. Bu durumda, Eldesiz Toplayıcı'nın en anlamlı biti 1 olduğunda (kaydırılamış ve kaydırılmış Elde'ler sırasıyla $n/2$ ve $n$'den büyük olduğunda) $YEN\_GRS_2$ 1 olur. Aksi durumda $Elde$ 1 üretemeyecek kadar büyük değildir, yani $YEN\_GRS_2$ 0 olur. Ayrıca, Algoritma 1'in 14. satırındaki çıkarma işlemini icra etmek için en alttaki kaydedicinin girişinin en anlamlı bitine daima 0 verilir. $YEN\_GRS_2$'nin 1 olması $Elde \geq n$ anlamına gelir, bu yüzden $Elde$'den $n$ çıkarılmalı, yani en anlamlı bit 0'a çevrilmelidir. Öte yandan, $REG\_IN_2 = 0$ ise $Elde$'de herhangi bir değişikliğe ihtiyaç yoktur, bu yüzden en anlamlı bit 0 olarak kalır.

Yardımcı işaretler $TRIG$ ve $CLR$'yi üretecek devre de Şekil 4.10'te bulunmaktadır. Aslında $CLR$, $TRIG$'in biraz geciktirilmiş versiyonudur, çünkü $TRIG$ işaretini kullanan kaydediciler en soldaki sayıcıların çıkışını sıfırlanmadan önce kaydetmelidirler. Şekil 4.10'teki tasarım $n$ bit girişler için genelleştirilebilir. Sayıcılar, kaydediciler ve ikili toplayıcılar $\log_2 n + 1$ bit olmalıdır. Ayrıca $2 \times (\log_2 n + 1)$ adet VE kapısı $CLR$ ve $PRE$ işaretlerinin üretimi için kullanılmalıdır. Yardımcı işaret üretiminde ise $log_2 n$ ardışık flip-flop kullanılacaktır. Neticede alan karmaşıklığı $O(\log n)$ olur.

# 5. DENEYSEL SONUÇLAR

Bu bölümde teklif edilen altı devrenin değerlendirilmesi yapılmıştır:

- **A**seknron **A**rtan **K**atar-uzunluklu **T**oplayıcı (**AAKT**),
- **A**seknron **A**rtan **K**atar-uzunluklu **Ç**arpıcı (**AAKÇ**),
- **S**enkron **A**rtan **K**atar-uzunluklu **T**oplayıcı (**SAKT**),
- **S**enkron **A**rtan **K**atar-uzunluklu **Ç**arpıcı (**SAKÇ**),
- **S**enkron **S**abit **K**atar-uzunluklu **T**oplayıcı (**SSKT**),
- **S**enkron **S**abit **K**atar-uzunluklu **Ç**arpıcı (**SSKÇ**).

Sonuçlar izleyen dört altbölümde gruplanmıştır. Bunların ilkinde, transistör seviyesinde benzetim sonuçları teklif edilen devrelerin pratik olarak uygun ve doğru biçimde çalıştığını göstermek için sunulmuştur. İkinci altbölümde, teklif edilen tasarımlar ile literatürdeki benzerlerinin transistör seviyesinde gecikne sonuçları karşılaştırılmıştır. 3. ve 4. altbölümlerde ise, teklif edilen toplayıcı ve çarpıcıların literatürdekilerle etraflıca karşılatırılması, sırasıyla münferit olarak ve bir yapay sinir ağları uygulamasında gerçekleştirilmiştir. Bu iki altbölümde, alan ve doğruluk göz önüne alınarak kapı seviyesinde benzetimler yapılmıştır.

## 5.1 Doğruluk için Transistör Seviyesinde Benzetimler

Benzetimle Cadence ortamında AMS 0.35µm CMOS teknolojisi ile yapılmıştır. Teklif edilen çarpıcı ve toplayıcılar 0.5ns'den 10ns'ye değişen 4 farklı bit genişliği için test edilmiştir. Giriş değerleri doğruluk açısından en kötü senaryo olan beklenen çıkış değerini 0.5 yapacak şekilde seçilmiştir. Giriş bit uzunlukları bütün benzetimlerde 8 olarak seçilmiştir.

İki adet başarım ölçütü göz önüne alınmıştır:

- Çıkış işaretinin bütünlüğü,

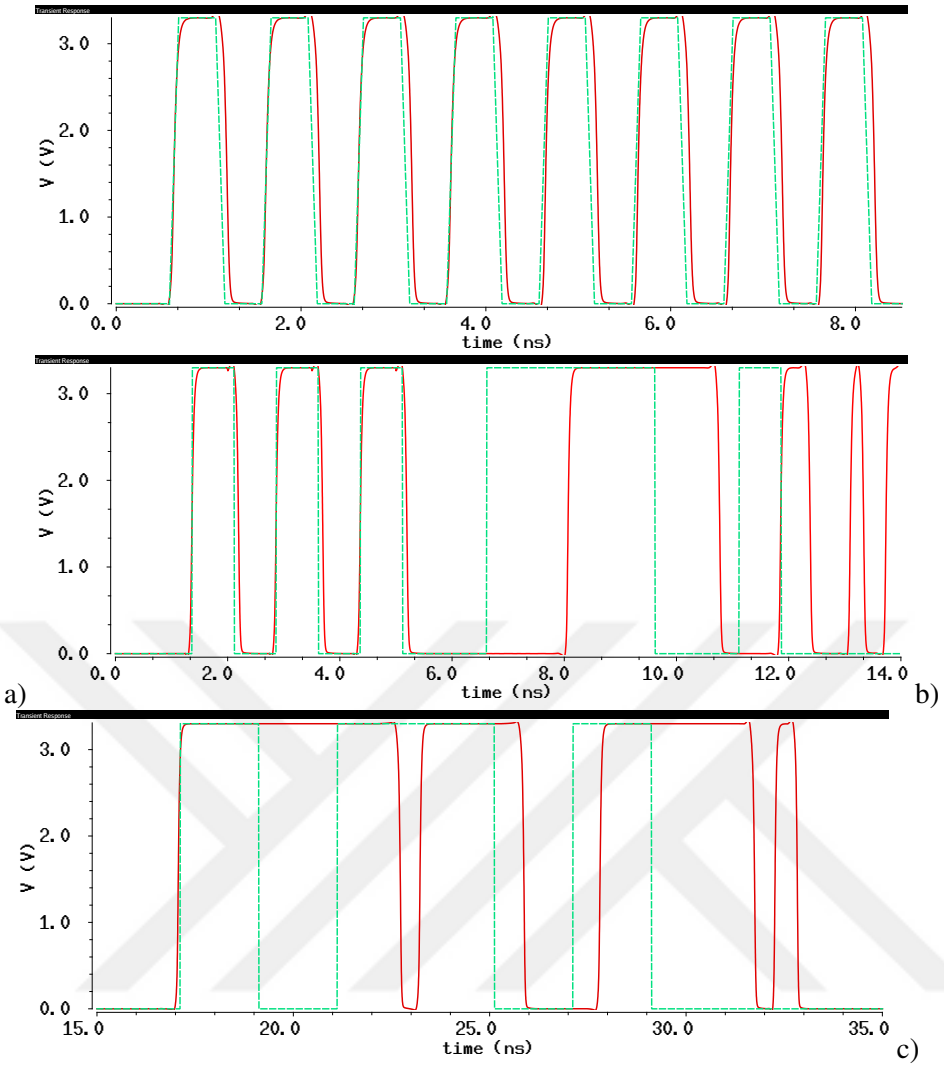
- Elde edilen çıkış değerinin doğruluğu

Bu ölçütleri ayrıntılı açıklamak amacıyla, 1ns'lik bit genişliği olan $1,0,1,1,0,0,0,1$ şeklinde beklenen bir çıkış katarı varsayılsın. Ayrıca benzetim sonucunda ise idealde sırasıyla 1ns, 2ns ve 1ns olması gereken 1 değerli bit genişliklerinin 1.1ns, 2.1ns ve 0.4ns olduğu varsayılsın. İlk ölçüt olan çıkış işaret bütünlüğü için, öncelikle mutlak sapmalar 0.1ns, 0.1ns ve 0.6ns; ardından da bağıl sapmalar da: 0.1ns/ 1ns, 0.1ns/2ns, ve 0.6ns/1ns olarak hesaplanır. Bağıl sapmaların ortalaması %25 olduğundan, işaret bütünlüğü %75 olur. İkinci ölçüt olan doğruluk için ise, çıkışta görülen bit genişlikleri görülmesi gerekenlere bölünür. İşaret bütünlüğü hesabında kullanılan örnek için oranlar sırasıyla 1.1ns/1ns, 2.1ns/2ns ve 0.4/1ns olur. Bu oranlar kendilerine en yakın tamsayıya yuvarlanır: 1, 2 ve 0. Bu tamsayıların toplanmasıyla çıkışta elde edilen değer 3/8 olarak hesaplanmış olur, bu da %75 doğruluk anlamına gelir.

Çizelge 5.1 teklif edilen devrelerin farklı bit genişlikleri için transistör seviyesinde doğruluk sonuçlarını vermektedir. SSKÇ haricindeki bütün devreler 1ns'ye eşit veya 1ns'den daha düşük bit genişliklerinde çalışabilirler, yani 1GHz'lik çalışma frekansı elde edilebilir. Aslında bu SSKÇ devresinin karmaşık bağlantılı ve geribeslemeli yapısından dolayı beklenen bir durumdur. Zamanlama hatalarından tamamen kaçınmak için devrenin istenen çalışma frekansına göre incelikle optimize edilmesi gerekir. Teklif edilen asenkron devrelerin basit ve gecikme bloğu tabanlı tasarımları sayesinde küçük bit genişlikleri için daha iyi performans gösterdikleri görülmektedir. Ancak bit genişliklerindeki artış daha fazla gecikme ihtiyacı ortaya çıkaracağından, bir noktadan sonra gecikme kontrolü mekanizması ile (bu durumda VDD ölçeklendirmesi) istenen gecikme miktarlarına ulaşılamayacaktır, ek eviricilere de ihtiyaç duyulacaktır. Bu durum bu devrelerin 2ns ve 10ns'de başarısız olmalarının sebebidir.

**Çizelge 5.1** : Teklif edilen çarpıcı ve toplayıcıların bütünlük ve doğruluk sonuçları (BÜT:Bütünlük, DOĞ:Doğruluk)

| Bit Genişliği | AAKT | | AAKÇ | | SAKT | | SAKÇ | | SSKT | | SSKÇ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BÜT | DOĞ | BÜT | DOĞ | BÜT | DOĞ | BÜT | DOĞ | BÜT | DOĞ | BÜT | DOĞ |
| 0.5ns | 87.8% | 100% | 91.8% | 100% | 31.8% | 37.5% | 0% | 0% | 56.9% | 50% | 0% | 0% |
| 0.75ns | 95.3% | 100% | 85.2% | 97.1% | 87.5% | 100% | 85.7% | 100% | 95.6% | 100% | 0% | 0% |
| 1ns | 92.5% | 100% | 58.4% | 60% | 90.5% | 100% | 89.2% | 100% | 95.5% | 100% | 0% | 0% |
| 2ns | N/A | N/A | N/A | N/A | 95.4% | 100% | 94.6% | 100% | 97.8% | 100% | 66.9% | 75% |
| 10ns | N/A | N/A | N/A | N/A | 99.1% | 100% | 98.9% | 100% | 98.2% | 100% | 94.8% | 100% |




**Şekil 5.1** : a) 0.5ns bit genişliği ile AAKT'nin, b) 0.750ns bit genişliği ile SAKT'nin, c) 2ns bit genişliği ile SSKÇ'nin transistör seviyesinde benzetim sonuçları. Kırmızı sürekli ve yeşil kesikli çizgiler sırasıyla gerçek ve beklenen çıkışları temsil etmektedir.

Şekil 5.1 üç farklı durum için çıkış işaretlerinin beklenen ve gerçek formlarını göstermektedir. Şekil Şekil 5.1.a)'da çıkış işareti iyi bir hizalanmaya sahipken, Şekil 5.1.b) ve c)'de istenmeyen iniş-çıkışlar gözükmektedir.

Girişlerin 8 bit seçilmesinin sebebi, nispeten basit bir durumda teklif edilen devrelerin pratik kullanımdaki potansiyelini göstermektir. Daha uzun bit katarları sistematik zamanlama tasarım stratejileri gerektireceğinden daha karmaşık devrelere ihtiyaç duyacaktır. Ancak, daha uzun katarlarda, oldukça tekdüze mimarileri sayesinde, asenkron tasarımların yine başarılı sonuçlara sahip olacağını tahmin etmek zor değildir. Bu durum tamamen ölçeklenebilir olan SSKT için de geçerlidir. Ancak, geri kalan

senkron tasarımlar, özellikle SSKÇ, daha uzun katarlar için daha da karmaşıklaşır, bu yüzden zamanlama hatalarına daha eğilimli olacaktır.

## 5.2 Gecikme için Transistör Seviyesinde Benzetimler

Bu altbölümde, teklif edilen tasarımlar ile literatürdeki stokastik ve ikili benzerlerinin gecikme benzetimlerinin sonuçları verilmiştir. Benzetim ortamı olarak yine Cadence ortamında AMS 0.35µm CMOS teknolojisi kullanılmıştır. Sonuçlar, daha isabetli bir şekilde incelenmek amacıyla, "Dahili Devre Gecikmesi (DDG)" ve "Katar Gecikmesi (KG)" olarak ikiye ayrılmıştır. 2. Bölüm'de tanımlanan bit genişliği ve katar uzunluğunun sırasıyla $W$ ve $n$ ile temsil edildiği varsayılsın. Geleneksel bir ölçüt olarak, DDG halihazırdaki devrenin en kötü propagasyon gecikmesidir. $W$'ya bağlı değildir, ancak ölçeklenebilir olmayan tasarımlarda $n$ ile artabilir. Öte yandan, KG spesifik olarak BKH/SH tasarımlar için bir gecikme ölçütüdür, ikili tasarımlar için uygulanamaz (N/A). Sıfır propagasyon gecikmesi varsayımıyla, giriş katarının başlangıcından çıkış katarının sonuna kadar geçen süre olarak tanımlanır. Neticede, sistemin toplam gecikmesi DDG ve KG'nin toplamı olur.

Çizelge 5.2 ve 5.3 teklif edilen toplayıcı ve çarpıcılar ile literatürdeki rakiplerinin gecikme benzetimi sonuçlarını göstermektedir. Çizelgeler 8 ve 16 giriş bit katar uzunlukları için DDG sonuçlarını içermektedir. Sonuçlardan en iyi DDG performansının teklif edilen asenkron devrelerde olduğu açığa çıkmaktadır. Ayrıca KG sonuçları $n$ ve $W$ cinsinden formulize edilmiştir. Son olarak, en sağdaki sütun da $n = 8$ ve $W = 2$ns durumunda devreler gerçek KG sonuçlarını vermektedir. KG için teklif edilen sabit katar uzunluklu tasarımlar en iyi sonuçlara sahip olduğu görülmektedir. BKH/SH tasarımları için, KG'nin toplam gecikmede aşırı derecede baskın olduğuna dikkat edilmelidir. Aslında,bu baskınlık herhangi anlamlı $n$ ve $W$ değerleri için geçerlidir. Bu da BKH/SH tasarımların analizi için sadece KG'nin göz önüne alınmasının tutarlı ve adil olacağı anlamına gelir.

### 5.2.1 Alan için kapı seviyesinde benzetimler

Karşılaştırmalarda; saat bölümü tabanlı [7], LFSR kullanan [5] ve PWM işaretleri kullanan [8] hatasız stokastik işlemler sunan çalışmalar ile, geleneksel ikili "Ripple Carry Adder" ve "Array Multiplier" devreleri göz önüne alınmıştır. Adil karşılaştırma yapmak adına, giriş ve çıkış işaretlerinin formları da hesaba katılmıştır. Teklif

**Çizelge 5.2** : Toplayıcıların gecikme karşılaştırması.

| Toplayıcılar | 16 bit için DDG (ns) | 8 bit için DDG (ns) | KG | $n = 8$ ve $W = 2$ns için KG |
|---|---|---|---|---|
| Ripple Carry Adder | 1.80 | 1.44 | N/A | N/A |
| [7] | 4.86 | 4.08 | $(n^2+n) \times W$ | 144 |
| [5] | 3.56 | 2.94 | $(n^2+n) \times W$ | 144 |
| [9] | 1.17 | 1.17 | $n \times W$ | 16 |
| AAKT | 0.15 | 0.15 | $2 \times n \times W$ | 32 |
| SAKT | 3.85 | 3.31 | $3 \times n \times W$ | 48 |
| SSKT | 1.17 | 1.17 | $n \times W$ | 16 |

**Çizelge 5.3** : Çarpıcıların gecikme karşılaştırması.

| Çarpıcılar | 16 bit için DDG (ns) | 8 bit için DDG (ns) | KG | $n = 8$ ve $W = 2$ns için KG |
|---|---|---|---|---|
| Array Multiplier | 3.39 | 2.67 | NA | NA |
| [7] | 4.70 | 3.92 | $(n^2+n) \times W$ | 144 |
| [5] | 3.40 | 2.78 | $(n^2+n) \times W$ | 144 |
| AAKÇ | 0.15 | 0.15 | $n^2 \times W$ | 128 |
| SAKÇ | 3.85 | 3.31 | $(n^2+n) \times W$ | 144 |
| SSKÇ | 5.19 | 4.29 | $2 \times n \times W$ | 32 |

edilen altı devrenin tamamının giriş ve çıkışları katar formundadır. Ancak, bu devrelerden senkron olanlar kendi içlerindeki sayıcı ve kaydedicilerle katardan-ikiliye çevrim yaptıkları için daha küçük devre boyutları ile ikiliden-katara hesaplama da yapabilirler (SAKT'nin 2. girişi hariç, ilgili yerde bu girişin ikili formata çevrilme maliyeti de göz önüne alınmıştır). Ancak [7] ve [5] ikili giriş ve katar çıkış formları kullanır; bu tasarımların katardan-katara hesaplama yapması için girişlerine sayıcı ve kaydediciler yerleştirilir. Bu durum aynı tasarımların katardan-katara hesaplama yaparken daha fazla alan harcamalarının sebebidir. Öte yandan, [8]'nin analog girişler kullanmasından ve bu girişlerden uygun çevrimlerin basitçe yapılamamasından dolayı, ayrı olarak değerlendirilmesi uygun görülmüştür.

**Çizelge 5.4** : Kullanılan lojik elemanların transistör sayıları (TS)

| | DEĞİL | VE-DEĞİL | VEYA-DEĞİL | AYRICALIKLI VEYA | D-Flip Flop | Half Adder | Full Adder |
|---|---|---|---|---|---|---|---|
| TS | 2 | 4 | 4 | 12 | 12 | 18 | 28 |

**Çizelge 5.5** : Toplayıcıların transistör sayısı karşılaştırması.

| | İkiliden-İkiliye | İkiliden-Katara | | | | Katardan-Katara | | | | | Analogdan-Katara |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Giriş Seviyeleri/ Katar Uzunluğu | Ripple Carry Adder | [7] | [5] | SAKT | SSKT | [7] | [5] | AAKT | SAKT | SSKT | [8] |
| 8 | 84 | 446 | 250 | 184 | 186 | 638 | 394 | 22 | 224 | 66 | 1637 |
| 16 | 112 | 602 | 346 | 230 | 226 | 842 | 538 | 38 | 280 | 66 | 3221 |
| 32 | 140 | 770 | 454 | 276 | 266 | 1058 | 694 | 70 | 336 | 66 | 6389 |
| 64 | 168 | 950 | 574 | 322 | 306 | 1286 | 862 | 134 | 392 | 66 | 12725 |
| 128 | 196 | 1142 | 706 | 368 | 346 | 1526 | 1042 | 262 | 448 | 66 | 25397 |
| 256 | 224 | 1346 | 850 | 414 | 386 | 1778 | 1234 | 518 | 504 | 66 | 50741 |

**Çizelge 5.6** : Çarpıcıların transistör sayısı karşılaştırması.

| | İkiliden-İkiliye | İkiliden-Katara | | | | Katardan-Katara | | | | | Analogdan-Katara |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Giriş Seviyeleri / Katar Uzunluğu | Array Multiplier | [7] | [5] | SAKÇ | SSKÇ | [7] | [5] | AAKÇ | SAKÇ | SSKÇ | [8] |
| 8 | 408 | 318 | 226 | 198 | 306 | 510 | 370 | 376 | 390 | 498 | 4784 |
| 16 | 688 | 430 | 314 | 262 | 394 | 670 | 506 | 1272 | 502 | 634 | 17984 |
| 32 | 1040 | 550 | 414 | 326 | 482 | 838 | 654 | 4600 | 614 | 770 | 69728 |
| 64 | 1464 | 678 | 526 | 390 | 570 | 1014 | 814 | 17400 | 726 | 906 | 274592 |
| 128 | 1960 | 814 | 650 | 454 | 658 | 1198 | 986 | 67576 | 838 | 1042 | 1089824 |
| 256 | 2528 | 958 | 786 | 518 | 746 | 1390 | 1170 | 266232 | 950 | 1178 | 4342304 |

Devrelerin Çizelge 5.4'deki değerler kullanılarak elde edilen transistör sayıları Çizelge 5.5 ve 5.6'de karşılaştırılmıştır. Farklı giriş seviyeleri değerlendirmeye alınmıştır; örneğin giriş seviyesinin 32 olduğu durum 5 ikili girişe veya 32 bitlik bir katar uzunluğuna tekabül eder. Table 5.5'da teklif edilen toplayıcıların bağlı bulundukları "ikiliden-katara" ve "katardan-katara" da diğerlerine bariz biçimde üstünlük sağladıkları görülmektedir. Genel karşılaştırmada ise ikili "Ripple Carry Adder", AAKT ve SSKT öne çıkmaktadır; AAKT ve SSKT sırasıyla küçük ve büyük giriş seviyelerinde en iyi sonuçları vermektedir. Ancak, SSKT'nin tamamen hatasız olmadığı düşünüldüğünde, ikili toplayıcının halen iddialı olduğu söylenebilir. Benzer çıkarımlar Çizelge 5.6'ten de elde edilebilir. Değinmeye değer bir diğer nokta da analogdan-katara hesaplamada transistör sayılarının aşırı fazla olmasıdır. Bunun sebebi [8]'de kullanılan işaret üreteçlerinin veya halka osilatörlerinin yüksek alan maliyetidir; bunlar evirici zincirleri ile gerçeklenmiştir. Hesaplamalarda, 1ns'lik bit genişliği seçilmiştir ki, bu Cadence ortamında AMS 0.35µm CMOS teknolojisi için eviricilerin geickme değerleri dikkate alındığında $n$ bitlik katar uzunlukları için $33 \times n$ eviricinin üreteçlerde kullanılması anlamına gelir. Alan maliyetinin bit genişliği azaltılarak düşürebileceğine dikkat edilmelidir. Ancak bunun bir alt sınırı vardır, bu sınır da transistör seviyesindeki benzetimler ve detaylı zamanlama analizleri ile belirlenebilir. Buradaki varsayımın oldukça adil olduğu düşünülmektedir. Transistör sayısını azaltmanın bir diğer yolu da osilatör topolojisini değiştirmektir.

Gecikme ve alan sonuçları birlikte yorumlanarak karşılaştırılan tasarımlar hakkında genel bir değerlendirme yapılabilir. Bit katarlı/stokastik tasarımlar arasında, sabit katar uzunluklu tasarımlar (SSKT-SSKÇ) hem alan hem de gecikme konusunda en iyi performansa sahip olduklarından öne çıkmaktadırlar.

Çizelge 5.7 bahsi geçen yöntemlerin niteliksel karşılaştırmasını yapmaktadır. Burada gecikme çıkış katarının süresi ile devrenin dahili gecikmesini içeren toplam hesaplama süresidir. Değerlendirmeye alan ve gecikmenin yanında ardışık bit katarlarını işlemeye ve çok-seviyeli tasarıma uygunluk kriterleri de eklenmiştir.

## 5.3 Yapay Sinir Ağları ile Değerlendirmeler

Yapay sinir ağları, çoğunlukla toplayıcılar ve çarpıcılardan oluştuğu ve mükemmel doğruluğa ihtiyaç duymadığı için, teklif eden tasarımların değerlendirileceği uygulama

**Çizelge 5.7** : Toplayıcı ve çarpıcıların niteliksel karşılaştırması.

| | Gecikme | Doğruluk | Alan | Ardışık İşleme | Çok-Seviyeli Tasarım |
|---|---|---|---|---|---|
| Geleneksel İkili | Çok İyi | Çok İyi | Vasat | İyi | Çok İyi |
| [7] | Vasat | Çok İyi | Kötü | Kötü | Vasat |
| [5] | Vasat | Çok İyi | Vasat | Kötü | Vasat |
| [8] | Vasat | Çok İyi | Kötü | Kötü | Kötü |
| AAKT/AAKÇ | Vasat | Çok İyi | Vasat | Kötü | Çok İyi |
| SAKT/SAKÇ | Vasat | Çok İyi | İyi | Kötü | Çok İyi |
| SSKT/SSKÇ | Vasat | İyi | İyi | Çok İyi | Çok İyi |
| Geleneksel Stokastik | Kötü | Kötü | Çok İyi | Çok İyi | Çok İyi |

alanı olarak seçilmiştir. El yazması rakamlardan oluşan PENDIGIT veritabanı, [13], önceden eğitilerek kullanılmıştır. Bu veritabanının 16 giriş karakteristiği vardır, ki bu sinir ağının giriş katmanında 16 düğüme karşılık gelir. Ayrıca sinir ağında 100 düğümlü bir gizli katman da bulunur. Çıkış katmanında 10 rakamı temsil etmek için 10 düğüm vardır. İkili toplayıcı ve çarpıcılarla yapılan geleneksel gerçekleme haricinde, her ikili-giriş değeri ile bu girişlerin ağırlıkları ikiliden-katara çarpıcılar ile çarpılmış, ardından katardan-katara toplayıcılar ile çiftler halinde toplanmıştır. Sonrasında, öncelikle katardan-ikiliye çevriciler (sayıcılar) kullanılarak, bir sonraki katmana kadarki bütün işlemler geleneksel ikili devreler ile yapılmıştır. Bu bölüm karşılaştırılan bütün çalışma/tekniklerde aynı olduğundan, alan masrafları hesaba dahil edilmemiştir.

Karşılaştırmalarda, dört farklı toplayıcı ve çarpıcı gerçeklemesi dikkate alınmıştır. Bunlardan ilki Çizelge 5.5 ve 5.6'te incelenenler arasında alan masrafı en düşük olan toplayıcı ve çarpıcıları teklif eden çalışmadır [5]. İkinci çalışma [9] teklif edilen SKST'ye çok benzer bir toplayıcı ve geleneksel stokastik çarpıcı (VE kapısı) kullanmaktadır. Üçüncüsü geleneksel "Ripple Carry Adder" ve "Array Multiplier" kullanır. Son tekniğin ilgili elemanları ise geleneksel stokastik toplayıcı (2:1 çoğullayıcı) ve çarpıcılardır (VE kapısı).

**Çizelge 5.8** : PENDIGIT veritabanı ile eğitilen bir sinir ağında kullanılan bütün toplayıcı ve çarpıcıların transistör sayıları (TS:Transistör Sayısı, YO:Yanlış Tasnif Oranı)

| Giriş Seviyeleri | [9] | | [5] | | SAKT-SAKÇ | | SSKT-SSKÇ | | Geleneksel İkili | | Geleneksel Stokastik | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TS | YO | TS | YO | TS | YO | TS | YO | TS | YO | TS | YO |
| 8 | 0.92M | 56.77% | 1.71M | 7.64% | 1.03M | 7.64% | 0.88M | 31.3% | 1.27M | 7.64% | 0.95M | 69.4% |
| 16 | 1.18M | 24.84% | 2.43M | 2.89% | 1.34M | 2.89% | 1.11M | 7.60% | 2.08M | 2.89% | 1.26M | 47.9% |
| 32 | 1.46M | 9.63% | 3.23M | 2.60% | 1.66M | 2.60% | 1.34M | 3.00% | 3.06M | 2.60% | 1.60M | 25.6% |
| 64 | 1.76M | 4.04% | 4.14M | 2.63% | 1.97M | 2.63% | 1.57M | 2.80} | 4.24M | 2.63% | 1.97M | 11.8% |
| 128 | 2.08M | 3.12% | 5.13M | 2.54% | 2.28M | 2.54% | 1.80M | 2.60% | 5.60M | 2.54% | 2.36M | 6.00% |
| 256 | 2.43M | 2.84% | 6.22M | 2.57% | 2.60M | 2.57% | 2.03M | 2.54% | 7.15M | 2.57% | 2.78M | 3.87% |

Geleneksel stokastik devreler rasgele dağılımlı giriş katarlarına ihtiyaç duyar. Bu yüzden her bir ikiliden-katara çarpıcıda 2'şer adet LFSR ve sayısal karşılaştırıcıya (comparator) ihtiyaç duyulur. [2]'de belirtildiği gibi, giriş katarları için 1 LFSR kullanmak, kaydırılmış katarlar arasındaki korelasyonun düşük olması sayesinde yeterli olacaktır. Ek olarak, her katardan-katara toplayıcı çoğullayıcının seçim girişindeki 0.5 değerli katarı üretmek için bir sayısal karşılaştırıcı daha gerekecektir, yine tek bir LFSR bütün toplayıcılar için yeterlidir.

Çizelge 5.8 ilgili sonuçları vermektedir. Teklif edilen gerçeklemelerin doğruluk ve alan açısından en iyi olduğu açıktır. Doğal olarak, stokastik sayı üreteçlerinin maliyeti dahil edilmeseydi, [9] ve geleneksel stokastik çok daha az transistör sayılarına sahip olacaktı, ancak bu adil bir karşılaştırma olmazdı.

# 6. SONUÇLAR VE TARTIŞMA

Bu çalışmada yeni bir hesaplama paradigması olarak “Bit Katarı Hesaplama (BKH)" sunulmuştur. BKH stokastik hesaplamanın alan avantajından ve geleneksek ikili hesaplamanın doğruluk avantajından faydalanmaktadır. BKH ile teklif edilen toplayıcı ve çarpıcı devreleri değerlendiren deneysel sonuçlar BKH’in alan ve doğruluk konusundaki verimliliğini tasdik etmektedir. Çalışmadan elde edilen kazanımların ilerletilmesi adına, teklif edilen tasarımların daha detaylı benzetimlerinin yapılması; ardından alan, gecikme, güç ve doğruluk ölçütlerinin zamanlama varyasyonları ile birlikte dikkate alınarak fabrikasyona geçirilmesi hedeflenmektedir. Bu çalışmada yapılan transistör seviyesindeki benzetimlerde, teklif edilen devrelerin 1ns’lik bit genişliği civarında çalışabildiği gözlenmiştir. Bu bit genişliğinin düşürülmesi de hedefler arasındadır. Daha genel ifadeyle, ana hedef spesifik olarak BKH için geliştirilecek yeni bir devre tasarım metodolojisi sunmaktır.

Bir başka odak noktası da teklif edilen devrelerin organik ve esnek (flexible) devreleri de içeren büyük alanlı elektronik uygulamalarında test edilmesidir. Büyük alanlı elektronik devreler nispeten az sayıda transistöre sahip olmaları gerektiğinden geleneksel ikili mantık devreleri bu uygulamalar için uygun değildir. Bu çalışmada teklif edilen BKH devrelerinin ise uygun olabileceği düşünülmektedir.

## KAYNAKLAR


[1] **Von Neumann, J.** (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components, *Automata studies*, *34*, 43–98.

[2] **Gaines, B.R.** (1967). Stochastic computing, *Proceedings of the April 18-20, 1967, spring joint computer conference*, ACM, s.149–156.

[3] **Alaghi, A. ve Hayes, J.P.** (2013). Survey of stochastic computing, *ACM Transactions on Embedded computing systems (TECS)*, *12*(2s), 92.

[4] **Ichihara, H.**, **Ishii, S.**, **Sunamori, D.**, **Iwagaki, T. ve Inoue, T.** (2014). Compact and accurate stochastic circuits with shared random number sources, *Computer Design (ICCD), 2014 32nd IEEE International Conference on*, IEEE, s.361–366.

[5] **Gupta, P.K. ve Kumaresan, R.** (1988). Binary multiplication with PN sequences, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *36*(4), 603–606.

[6] **Alaghi, A. ve Hayes, J.P.** (2013). Exploiting correlation in stochastic circuit design, *Computer Design (ICCD), 2013 IEEE 31st International Conference on*, IEEE, s.39–46.

[7] **Jenson, D. ve Riedel, M.** (2016). A deterministic approach to stochastic computation, *Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on*, IEEE, s.1–8.

[8] **Najafi, M.H.**, **Jamali-Zavareh, S.**, **Lilja, D.J.**, **Riedel, M.D.**, **Bazargan, K. ve Harjani, R.** (2017). Time-Encoded Values for Highly Efficient Stochastic Circuits, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *25*(5), 1644–1657.

[9] **Lee, V.T.**, **Alaghi, A.**, **Hayes, J.P.**, **Sathe, V. ve Ceze, L.** (2017). Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing, *2017 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, s.13–18.

[10] **Brown, B.D. ve Card, H.C.** (2001). Stochastic neural computation. I. Computational elements, *IEEE Transactions on computers*, *50*(9), 891–905.

[11] **Ting, P. ve Hayes, J.P.** (2017). Eliminating a hidden error source in stochastic circuits, *2017 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, IEEE, s.1–6.

[12] **Vahapoglu, E. ve Altun, M.** (2016). Accurate Synthesis of Arithmetic Operations with Stochastic Logic, *VLSI (ISVLSI), 2016 IEEE Computer Society Annual Symposium on*, IEEE, s.415–420.

[13] **Alpaydin, E. ve Alimoglu, F.** (1998). Pen-based recognition of handwritten digits data set, *University of California, Irvine, Machine Learning Repository. Irvine: University of California*.

## ÖZGEÇMİŞ

**Ad Soyad:** Ensar Vahapoğlu

**Doğum Tarihi ve Yeri:** 21.08.1992 - Trabzon

**E-Posta:** vahapoglu@itu.edu.tr

### ÖĞRENİM DURUMU:

- **Lisans:** 2015, İstanbul Teknik Üniversitesi, Elektrik-Elektronik Fakültesi, Elektronik ve Haberleşme Mühendisliği Bölümü.

### MESLEKİ DENEYİMLER VE ÖDÜLLER:

- 2016'dan günümüze kadar İstanbul Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği Bölümü'nde Araştırma Görevlisi olarak çalıştı.

### YÜKSEK LİSANS TEZİNDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- Vahapoglu E., Altun M., 2016. Accurate Synthesis of Arithmetic Operations Using Stochastic Logic. *IEEE Computer Society Annual Symphosşum on VLSI*, July 11-13, 2016, Pittsburgh, PA, USA.

- Vahapoglu E., Altun M., 2018. From Stochastic to Bit Stream Computing: Accurate Implementation of Arithmetic Circuits and Applications in Neural Networks, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* (under review).