

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KONUMSAL KORELASYON KULLANARAK
VARYASYONLARI BASKILAYAN GECİKME TEMELLİ
DONANIMSAL TRUVA ATI TESPİTİ

FATMA NUR ESİRCİ

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN: DR. ÖĞR. ÜYESİ ALP ARSLAN BAYRAKÇI

AĞUSTOS 2024

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KONUMSAL KORELASYON KULLANARAK
VARYASYONLARI BASKILAYAN GECİKME
TEMELLİ DONANIMSAL TRUVA ATI TESPİTİ

FATMA NUR ESİRCİ

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN: DR. ÖĞR. ÜYESİ ALP ARSLAN BAYRAKÇI

AĞUSTOS 2024

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

**SUPPRESSING VARIATIONS BY SPATIAL
CORRELATIONS TO AVAIL DELAY-BASED
HARDWARE TROJAN DETECTION**

FATMA NUR ESİRCİ

A THESIS OF DOCTORATE
DEPARTMENT OF COMPUTER ENGINEERING

ADVISOR: ASST. PROF. DR. ALP ARSLAN BAYRAKÇI

AUGUST 2024

DOKTORA JÜRİ ONAY FORMU

GTÜ Lisansüstü Eğitim Enstitüsü Yönetim Kurulunun 24/06/2024 tarih ve 2024/32 sayılı kararıyla oluşturulan jüri tarafından 26/06/2024 tarihinde tez savunma sınavı yapılan Fatma Nur Esirci'nin tez çalışması Bilgisayar Mühendisliği Anabilim Dalında DOKTORA tezi olarak kabul edilmiştir.

JÜRİ

ÜYE

(TEZ DANIŞMANI) : Dr. Öğr. Üyesi Alp Arslan Bayrakçi

ÜYE

: Doç. Dr. Engin Afacan

ÜYE

: Prof. Dr. Didem Gözüpek Kocaman

ÜYE

: Dr. Öğr. Üyesi Sanem Yılmaz Arslan

ÜYE

: Dr. Öğr. Üyesi Arzu Kakışım

ONAY

Gebze Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Yönetim Kurulunun
...../...../..... tarih ve/..... sayılı kararı.

İMZA/MÜHÜR



*Sevgili Anneciğim
Neriman Hanım'a,
özlem ve minnetle...*

ÖZET

Donanımsal Truva atları (DTA), entegre devrelerde güvenliği tehdit eden gizli bileşenlerdir. Truva atlarının tespiti, özellikle üretim sonrası testler sırasında ve süreç varyasyonlarının etkisi altında büyük zorluklar yaratmaktadır.

Üretim sırasında oluşan ve istenmeyen etkilere neden olan süreç varyasyonları, devre elemanlarının davranışında önemli değişikliklere neden olarak Truva atı tespitini zorlaştırmaktadır. Truva atları, devrenin normal işleyişini bozmadan gizli kalabildikleri için tespit edilmeleri zordur. Truva atlarının küçük boyutlu ve düşük etkili olması, geleneksel test yöntemleri ile fark edilmelerini engellemektedir.

Gecikme tabanlı yöntemler, Truva atlarının devrede neden olduğu yol gecikmesi artışlarını tespit etmek için etkili bir yoldur. Bu yöntemler, üretim sonrası testlerde uygulanabilir ve ek donanım gerektirmezler. Ancak, süreç varyasyonlarından kaynaklanan yanlış pozitif sonuçlar, gecikme tabanlı yöntemlerin etkinliğini azaltmaktadır.

Bu tez çalışmasında, Truva atlarını tespit etmek için yol gecikmesi tabanlı yeni bir yöntem öneriyoruz. Yöntemimizde süreç varyasyonlarının etkilerini ortadan kaldırmak için konumsal korelasyonlardan yararlanmaktayız. Yöntem, devre içinde şüpheli yolları belirleyerek ve bu yolların gecikme oranlarını analiz ederek Truva atı tespiti yapmaktadır. Tüm gecikme modelleri, hem yonga-içi hem de yongalar arası varyasyon bileşenlerini dikkate alan ayrıntılı ve gerçekçi bir varyasyon modeli kullanmaktadır.

Yöntemimiz, yol çiftlerini konumsal korelasyonlar kullanarak seçer ve bu yol çiftleri sayesinde varyasyon etkileri minimize edilir. Literatürdeki gecikme tabanlı yaklaşımlar arasında türünün ilk örneğidir. Konumsal korelasyondan yararlanmanın, varyasyonun gecikmeye olan etkisini efektif bir şekilde azaltabileceğini gösteren ilk çalışmadır. Geliştirilen Truva atı tespit yöntemi, küçük boyutlu olarak tanımlanabilen Truva atlarını bile yüksek doğrulukla tespit edebilme yeteneğine sahiptir. Ayrıca, altın model gerektirmeyen ve üretim aşamasında ek donanım değişikliği gerektirmeyen bir teknik sunmaktadır.

Truva atlarının farklı türleri vardır. Çalışmamızda tespit edilebilmesi için, yol gecikmesini etkilemesi gereken Truva atlarına odaklandık. Kullandığımız varyasyon modeli çeşitli parametrelere bağlıdır ve daha karmaşık varyasyon senaryolarında performansta değişiklikler gözlemlenebilir. Yöntemin doğruluğu, test edilen yonga sayısına ve varyasyon parametrelerinin doğruluğuna bağlıdır.

Anahtar Kelimeler: Donanımsal Truva Atı, Truva Atı Tespiti, Yan-Kanal Analizi, Yol Gecikmesi, Konumsal Korelasyon, Gerçekçi Varyasyon Modeli.

ABSTRACT

Hardware Trojans (HTs) are hidden components that pose significant security threats to integrated circuits. Detecting these Trojans, especially during post-manufacturing tests and under the effect of process variations, presents considerable challenges.

Process variations, which occur during manufacturing and cause unintended effects, lead to significant changes in the behavior of circuit components, making Trojan detection challenging. HTs are difficult to detect as they can remain hidden without disrupting the normal operation of the circuit. Their small size and low impact often evade traditional testing methods.

Delay-based methods are effective in detecting the path delay increases caused by HTs in the circuit. These methods are applicable in post-manufacturing testing and do not require additional hardware. However, false positives due to process variations can reduce the effectiveness of delay-based methods.

In this thesis, we propose a novel delay-based method to detect HTs. We leveraged spatial correlations to eliminate the effects of process variations in our method. The method identifies suspected paths within the circuit and analyzes their path-delay ratios to detect HTs on suspected paths. We used a detailed and realistic variation model that considers both intra-chip and inter-chip variation components.

Our method selects path pairs using spatial correlations, thereby minimizing the effects of variations. Our method is the first of its kind among delay-based methods in the literature. It is the first to prove that exploiting spatial correlation can effectively reduce the effect of variation. The developed HT detection technique is capable of detecting even small HTs with high accuracy. Additionally, it offers a technique that does not require a golden model and does not necessitate any additional hardware modifications during production.

There are different types of HTs. In our study, we focus on HTs that must affect path delays. Our considered variation model depends on several parameters and the performance may vary in more complex. The accuracy of the method depends on the number of chips tested and the accuracy of the variation parameters.

Keywords: Hardware Trojan, Hardware Trojan Detection, Side-channel Analysis, Delay-based, Spatial Correlation, Realistic Variation Model.

TEŞEKKÜR

Doktora eğitimim boyunca bana destek vererek, tez yazımı sürecini tamamlamamda emeği geçen başta değerli tez danışmanım Dr. Öğr. Üyesi Alp Arslan Bayrakçı olmak üzere tez izleme jürimde yer alarak bilgi birikimini benimle paylaşan Prof. Dr. Didem Gözüpek Kocaman ve Doç. Dr. Engin Afacan'a şükranlarımı sunarım. Tez savunma jürimde yer alan Dr. Öğr. Üyesi Arzu Görgülü Kakışım ve Dr. Öğr. Üyesi Sanem Arslan Yılmaz'a göstermiş oldukları desteklerden dolayı en içten teşekkürlerimi sunarım. Bütün çalışmam boyunca yanımda olan ve çeşitli sağlık sorunlarıyla baş etmemde yardımcı olan eşim Haluk Oral'a, kardeşim Şule Nur Esirci'ye ve çocuklarım Gufi, Nane, Sani ve Zeytin'e koşulsuz destekleri için minnetarım.

TÜBİTAK Bilim İnsanı Destek Programları Başkanlığı'na (BİDEB), 2211-C Yurt İçi Öncelikli Alanlar Lisansüstü Burs Programı kapsamında verdikleri destekler için teşekkür ederim.



İÇİNDEKİLER

	<u>Sayfa</u>
ÖZET	v
ABSTRACT	vi
TEŞEKKÜR	vii
İÇİNDEKİLER	ix
SİMGELER ve KISALTMALAR DİZİNİ	x
ŞEKİLLER DİZİNİ	xii
TABLolar DİZİNİ	xiii
1. Giriş	1
1.1. Tümüleşik Devreler	1
1.1.1. Devrelerin Zayıf Yönleri	2
1.2. Truva Atları	6
1.2.1. Etkileri	7
1.2.2. Sınıflandırması	8
1.3. Problem Tanımı ve Kapsamı	12
1.3.1. Problemin Önemi	15
1.3.2. Yaklaşımımız	16
1.4. Süreç Varyasyonu	18
1.5. Tezin Organizasyonu	22
2. Literatür Özeti	24
2.1. İlgili Çalışmalar: DTA Tespiti Yöntemleri	24
2.1.1. Tersine Muhendislik Yöntemleri	24
2.1.2. Mantıksal Test Yöntemleri	26
2.1.3. Yan Kanal Analizi Yöntemleri	28
2.1.4. Çalışma Sırasında Tespit Eden Yöntemler	31
2.2. İlgili Çalışmalar: Yol Gecikmesi Kullanan Yöntemler	32
2.2.1. Kullanılan Devreler	35
2.2.2. Karşılaşılan Zorluklar	36
3. Yönteme Hazırlık	38
3.1. Devre Çizgesi	38
3.2. Gerçekçi Varyasyon Modelimiz	40
3.3. Gecikme Ölçümü için Stokastik Lojik Efor Modeli	42
3.4. Kullanılan Truva Atı Modeli	43
3.5. Notasyonlar	45
3.6. Hassas Stokastik Lojik Efor Yöntemi	46
3.7. Yol Gecikmesi Ölçümlerinin Doğrulanması	49

4. Konumsal Korelasyonlardan Yararlanarak Yol Gecikmesi ile Truva Atı Tespiti	51
4.1. Korelasyon Katsayısına Dayalı Truva Tespit Yöntemi (KKY)	52
4.1.1. Aşama-0: Kapı Gecikmesi Hesabı	52
4.1.2. Aşama-1: Şüpheli Yol (P_{susp}) Elde Etme	53
4.1.3. Aşama-2: İlişkili Yol Adayları Kümesi (π_{corr}) Oluşturma	54
4.1.4. Aşama-3: Yol İkili ($\langle P_{susp}, P_{corr} \rangle$) Seçimi	56
4.1.5. Aşama-4: Üretim Sonrası Truva Atı Tespiti	57
4.2. Paylaşılan İzgaraların Oranına göre Truva Atı Tespit Yöntemi (PIY)	59
4.2.1. Yol İkili ($\langle P_{susp}, P_{corr} \rangle$) Seçimi	61
4.3. Rastgele Uyarlanabilir Yöntem ile Truva Atı Tespiti (RUY)	63
4.3.1. İlişkili Yol Adayları Kümesi (π_{corr}) Oluşturma	63
4.3.2. Yol İkili ($\langle P_{susp}, P_{corr} \rangle$) Seçimi	65
4.3.3. Üretim Sonrası Testlerin Hızlandırılması	68
4.3.4. Üretim Sonrası Truva Atı Tespiti	70
5. Deneysel Sonuçlar	73
5.1. Korelasyon Katsayısına Dayalı Truva Tespit Yöntemi Sonuçları	73
5.2. Paylaşımlı İzgaraların Oranına göre Truva Atı Tespit Yöntemi Sonuçları	75
5.3. Rastgele Uyarlanabilir Yöntem ile Truva Atı Tespit Sonuçları	78
5.4. Truva Atının Tetikleyici Kısımının Tespit Edilebilirliği	83
5.5. Ardışık Devrelerde Truva atı Tespitine Dair Sonuçlar	85
6. Yol Gecikmesi ile Truva Atı Tespitinin FPGA	
Tabanlı Doğrulaması	86
6.1. Yol Gecikme Ölçüm Devresi	86
6.2. Gecikme Değeri Üzerindeki Süreç Varyasyonu Etkisi	87
6.3. Truva Atının Gecikmeye Etkisi	90
6.4. Varyasyonları Bastırmak için Halka Osilatörü Kullanımı	91
6.5. Truva Atının Boyut Değişimi	94
7. Tartışmalar	98
7.1. Gelecek Çalışmalar	98
KAYNAKLAR	100
ÖZGEÇMİŞ	106
TEZ ÇALIŞMASI KAPSAMINDA YAPILAN YAYINLAR	107

SİMGELER ve KISALTMALAR DİZİNİ

μ	: Orta Değer
σ	: Standart Sapma
V_t	: Eşik Voltajı
L_{eff}	: Kapı uzunluğu
T_{ox}	: Oksit Kalınlığı
n_p	: Düğüm p
$e_{p,q}$	: Kenar p, q
P_{susp}	: Şüpheli Yol
$\hat{P}_{susp}$	: Truva Atı içeren Şüpheli Yol
π_{corr}	: İlişkili Yol Adayları
P_{corr}	: İlişkili Yol
R	: Yol Gecikmesi Oranı
$\hat{R}$	: Truva Atlı Yol Gecikmesi Oranı
ASIC	: Uygulamaya Özgü Tümdevre
BDT	: Bilgisayar Destekli Tasarım
ÇBÖT	: Çok Büyük Ölçekte Tümleşim
DTA	: Donanımsal Truva Atı
EDA	: Elektronik Tasarım Otomasyonu
FPGA	: Kullanıcı Programlamalı Kapı Dizisi
FM	: Fikri Mülkiyet
GTÜ	: Gebze Teknik Üniversitesi
HO	: Halka Osilatörü
HSLE	: Hassas Stokastik Lojik Efor
OTDÜ	: Otomatik Test Deseni Üretimi
RTL	: Yazmaç Aktarım Seviyesi
SAT	: Tatmin Edicilik Testi
TEM	: Taramalı Elektron Mikroskobu
SLE	: Stokastik Lojik Efor
YKA	: Yan Kanal Analizi

ŞEKİLLER DİZİNİ

	<u>Sayfa</u>
Şekil 1.1: Çok Katmanlı Tümleşik Devre Gösterimi [1].	1
Şekil 1.2: ÇBÖT Endüstrisi Üretim Modelleri [2].	2
Şekil 1.3: Tümleşik Devrelerin Üretim Aşamaları	3
Şekil 1.4: Tümleşik Devrelerin Yaşam Döngüsü [3].	4
Şekil 1.5: Katmanlar Üzerinde Truva Atının Gösterimi [1] .	7
Şekil 1.6: Donanımsal Truva Atı Modelinin Genel Gösterimi.	7
Şekil 1.7: Donanımsal Truva Atlarının Sınıflandırılması.	8
Şekil 1.8: c7552 devresindeki bir yola ait truva atlı ve atsız gecikme değerleri.	13
Şekil 1.9: Yol çiftlerinin gecikme oranları (a) rastgele konumlarda bulunan, (b) birbirlerine yakın konumlarda bulunan	14
Şekil 1.10: Levha örneği: TSMC 100nm yonga üretimi.	18
Şekil 1.11: Süreç varyasyonlarının sınıflandırılması.	19
Şekil 1.12: Yongalar-arası ve yonga-içi değişimlerin frekans parametresi üzerinde gösterimi.	20
Şekil 2.1: Truva atı tespit yöntemlerinin sınıflandırılması.	24
Şekil 2.2: (a):Tasarımdaki Görünüm , (b):Elektron mikroskopuyla görünüm.	25
Şekil 2.3: ATPG Araçlarının Çalışma Süreci.	27
Şekil 2.4: YKA Yöntemlerinin Çalışma Süreci.	29
Şekil 2.5: Çalışma sırasında DTA tespitinin gösterimi.	31
Şekil 2.6: DEĞİL Kapısına ait Yayılma Gecikmesinin Ölçümü.	32
Şekil 3.1: Devre Çizgesi Oluşturma.	38
Şekil 3.2: Devre Çizgesi Oluşturma Aşamaları.	39
Şekil 3.3: Dördün Ağaç Yapısı.	41
Şekil 3.4: Kullanılan Donanımsal Truva Atı Modeli.	44
Şekil 3.5: Truva Atının Devreye Eklenme Çeşitleri.	45
Şekil 3.6: L2H, H2L ve ortalama gecikme değerleri kullanıldığında elde edilen gecikme oran dağılımları.	47
Şekil 3.7: L2H ve H2L Gecikme Ölçümlerinin Gösterimi.	48
Şekil 3.8: Truva Atının Tetikleyici Kısmı olan Yoldışı Parametrelerin Etkisi.	49
Şekil 3.9: Simulasyon ve SPICE sonuçlarının Karşılaştırılması.	50
Şekil 4.1: Üretim öncesinde yapılması gereken aşamaların gösterimi.	51
Şekil 4.2: Kapı Gecikmesi Hesabı için Yapılanlar.	53
Şekil 4.3: Şüpheli Yol Seçimi.	54
Şekil 4.4: Tüm devre yerleşiminin ızgaralara bölünmesi.	55
Şekil 4.5: İlişkili Yol Adayları Kümesi Oluşturma.	55
Şekil 4.6: İlişkili Yol Adayları Kümesi Oluşturma.	56
Şekil 4.7: Üretim Sonrasında Truva Atı Tespiti.	58
Şekil 4.8: DTA içermeyen (yeşil) ve DTA eklenmiş (kırmızı) Gecikme oran Dağılımlarının Gösterimi.	59
Şekil 4.9: DTA içermeyen (yeşil) ve DTA eklenmiş (kırmızı) Gecikme oran Dağılımlarının Gösterimi.	60
Şekil 4.10: İlişkili yol adaylarının korelasyon katsayılarının gösterimi.	61

Şekil 4.11:	İlişkili yol adayları seçiminde kullanılan değerlerin karşılaştırması	62
Şekil 4.12:	İlişkili Yol Adayları Kümesi Yol Oluşturma.	63
Şekil 4.13:	Yol İkili Oluşturma.	66
Şekil 4.14:	Yol çifti sonuçları; (a): <i>DD</i> negatif , (b): <i>DD</i> pozitif.	67
Şekil 4.15:	HSLE modelinin SPICE ile doğrulanması.	70
Şekil 4.16:	Üretim Sonrasında Truva Atı Tespiti.	71
Şekil 4.17:	c3540 devresinden çıkarılmış yol ikilileri: (a): Truva atı tespit edilemedi, (b): Truva atı tespit edildi.	72
Şekil 5.1:	Aşama-3 hızlanması: PIY yönteminin KKY'ye göre.	79
Şekil 5.2:	Tüm Aşamaların Hızlanması: PIY yönteminin KKY'ye göre.	79
Şekil 6.1:	Yol Gecikmesi Ölçümünü Yapan Sonlu Durum Makinesi Tasarımı.	86
Şekil 6.2:	8 Aynı Yolun FPGA üzerinde Farklı Konumlara Yerleşimi.	88
Şekil 6.3:	20 FPGA üzerinde ölçülen yol gecikmeleri.	89
Şekil 6.4:	Yarısına Truva Atı Eklenmiş FPGA'ler üzerinde Ölçülen Yol Gecikmeleri.	90
Şekil 6.5:	Genel Bir Halka Osilatör Devresi.	91
Şekil 6.6:	Test Düzeneklerinin Genel Görünümü.	92
Şekil 6.7:	50 Kopya Yol ve HO'nün FPGA Yerleşiminin Gösterimi.	92
Şekil 6.8:	Yarısına Truva Atı Eklenmiş FPGA'ler üzerinde Yol Gecikme Oranları.	93
Şekil 6.9:	Dörtte Birine Truva Atı Eklenmiş FPGA'ler üzerinde Yol Gecikme Oranları.	93
Şekil 6.10:	Truva Atı Yük Kısmı: İki Adet Seri Bağlı XOR Kapısı.	95
Şekil 6.11:	Truva Atı Yük Kısmı: Üç Adet Seri Bağlı XOR Kapısı.	96
Şekil 6.12:	Farklı Truva Atı Boyutları için Gecikme Oranı Histogramları.	96

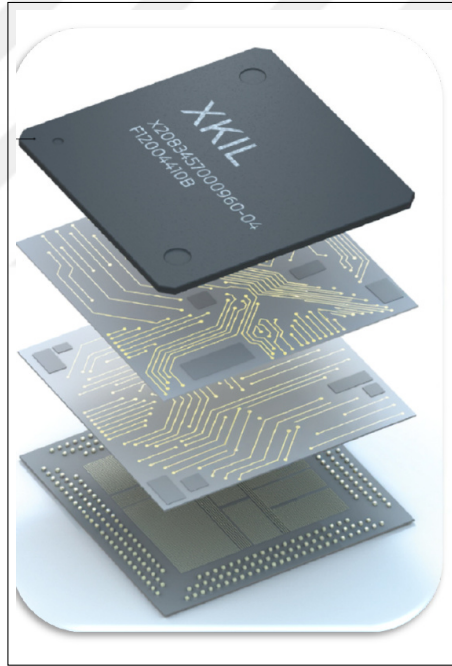
ÇİZELGELER DİZİNİ

	<u>Sayfa</u>
Tablo 1.1: Kritik Bir Sistemde Durum Tabanlı Truva Atı Örneği.	9
Tablo 2.1: YKA yöntemlerinin Karşılaştırılması.	30
Tablo 2.2: Yol Gecikmesi Yöntemlerinin Karşılaştırması.	33
Tablo 4.1: Minimum Küme Kapsama Yönteminin Çalışmasına Dair Örnek.	69
Tablo 5.1: KKY Yöntemine ait Sonuçlar.	74
Tablo 5.2: KKY Yönteminin Zaman Kullanımı.	74
Tablo 5.3: KKY ve PIY Sonuçlarının Karşılaştırılması.	77
Tablo 5.4: Rastgele Uyarlanabilir Yöntemine ait Sonuçlar.	80
Tablo 5.5: Aşamaların Çalışma Zamanları (saniye cinsinden).	82
Tablo 5.6: Truva Atının Tetikleyici Kısımına ait Sonuçlar.	84
Tablo 5.7: Trust-Hub Ardışık Devrelerine ait Sonuçlar.	85
Tablo 6.1: FPGA üzerinde 8 Farklı Konumdaki Yol Gecikmeleri (nano saniye).	89
Tablo 6.2: Çift Model Yöntemlerinin Test Sonuçları.	97

1. Giriş

1.1. Tümleşik Devreler

Tümleşik devreler (ing. IC), elektronik devrelerin bir yonga üzerine miniyatürleştirilmiş halidir ve günümüz teknolojisinin temel taşlarını oluşturmaktadırlar. Bu devreler, bilgisayarlardan cep telefonlarına, endüstriyel makinelerden tüketici elektroniğine kadar geniş bir yelpazede kullanılmaktadırlar. Tümleşik devreler, pasif (dirençler, kapasitörler gibi) ve aktif (transistörler gibi) bileşenlerin bir araya getirilerek oluşturduğu karmaşık yapılar olarak ifade edilebilirler. Bu bileşenler, yonga üzerine entegre edilerek elektronik cihazların çeşitli fonksiyonlarını yerine getirmelerini sağlamaktadırlar.



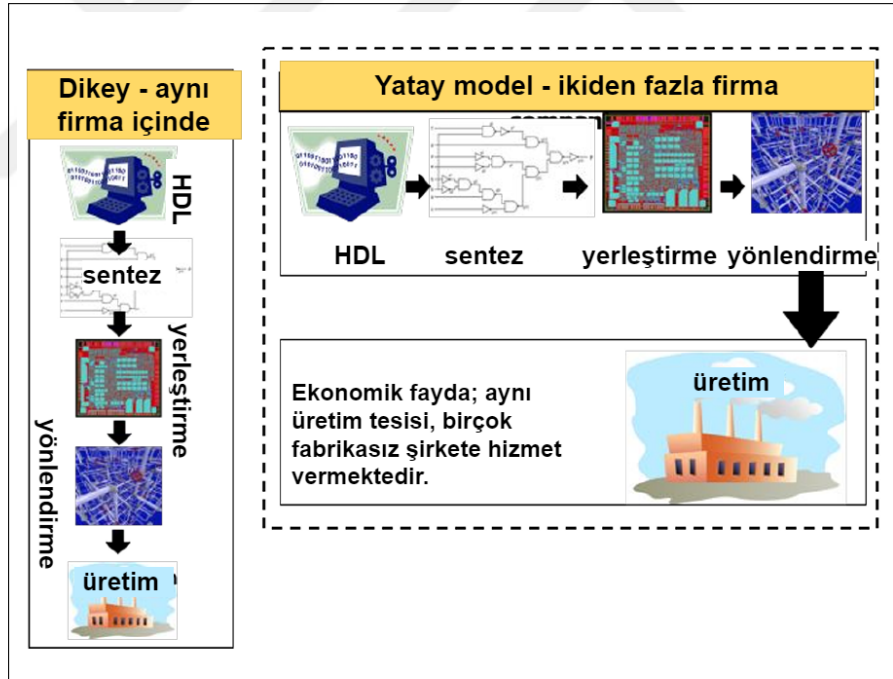
Şekil 1.1: Çok Katmanlı Tümleşik Devre Gösterimi [1].

Çok Büyük Ölçekli Tümleşim (ing. Very Large Scale Integrated Circuit) tarihçesi, ilk tümleşik devrenin geliştirildiği 1950'lerin sonlarında başlamıştır. İlk tümleşik devreler sadece birkaç transistör içeriyordu ve genellikle askeri ve havacılık uygulamalarında kullanılıyordu. 1960'larda, hesap makineleri, saatler ve erken büyük bilgisayarlar gibi ticari uygulamalarda kullanılmıştır. 1970'lerde Carver Mead, on binlerce transistörü tek bir silikon yonga üzerine başarıyla entegre ederek ÇBÖT (ing. VLSI) geliştirmiştir [2].

Günümüzde ÇBÖT teknolojisi, akıllı telefonlar, dizüstü bilgisayarlar, arabalar, uçaklar ve tıbbi cihazlar gibi birçok alanda kullanılmaktadır. Modern ÇBÖT çipleri, milyarlarca transistör içermekte ve sadece birkaç santimetre boyutunda yer kaplamaktadırlar.

Tümleşik devreler, yaklaşık 0.000005 ila 0.1 mm kalınlığında katmanlardan oluşmakta ve yarı iletken alt tabaka üzerine tek tek inşa edilmektedirler. Bir çipte en az 30 katman bulunmaktadır (Şekil 1.1). Çip üzerindeki farklı elektriksel bileşenleri oluşturmak, her katmanda n-tipi ve p-tipi alanların tam olarak nerede bulunacağını belirlemekle ilgilidir. Her katmanda istenen alanları oluşturmak için ışığa duyarlı malzemeler ile çeşitli kimyasal ve fiziksel reaksiyonlardan yararlanarak; aşındırma, yeni katman ekleme, katmanları birleştirme gibi yöntemler kullanılmaktadır.

1.1.1. Devrelerin Zayıf Yönleri

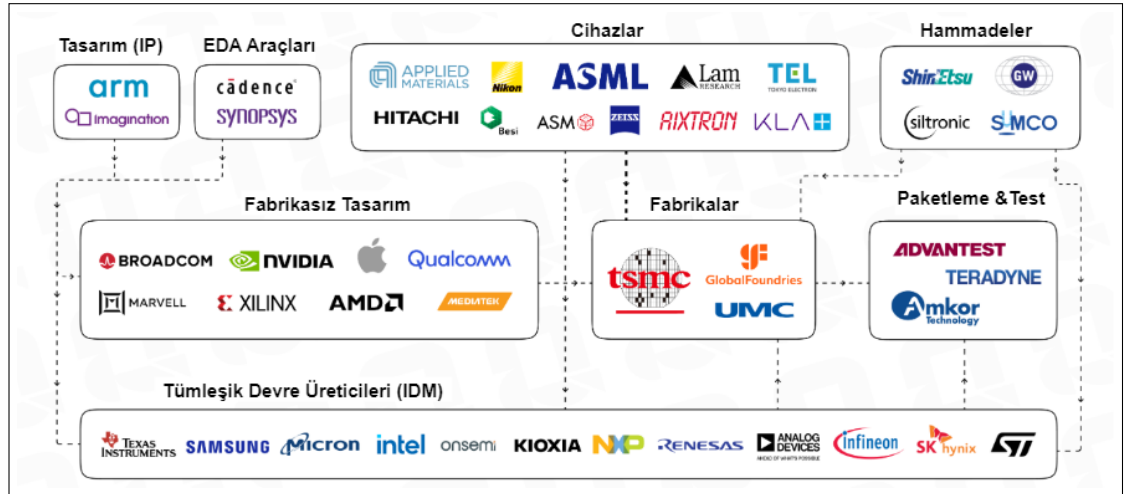


Şekil 1.2: ÇBÖT Endüstrisi Üretim Modelleri [2].

Teknolojik gelişmeler, tümleşik devrelerde kullanılan transistör boyutlarının giderek küçülmesine, aynı zamanda devrelerin daha fazla işlevsellik kazanmalarına olanak sağlamıştır. Bu süreç, mikroelektronik alanındaki yenilikler sayesinde mümkün olmakta, böylece devreler daha az yer kaplayarak daha kompleks görevleri yerine getire-

bilmektedir. Ancak transistör boyutunun küçülmesi tümleşik devrelerin performansını ve verimliliğini artırsa da, yüksek yoğunluklu devre tasarımları nedeniyle ısınma, sinyal bozulması ve elektriksel parazitler gibi teknik zorlukları da beraberinde getirmektedir [4–6]. Bu zorluklar, güvenlik ve dayanıklılık açısından yeni zayıf noktalar ortaya çıkarmaktadır. Özellikle, küçük ölçekli tasarımlar arasındaki yoğun etkileşim, devrelerin hassas olduğu çevresel etmenlere karşı daha savunmasız hale gelmesine yol açmaktadır. Tümleşik devrelerin zayıf yönleri, genellikle üretim sürecindeki kusurlar, tasarım hataları ve dış etkenlere olan hassasiyetlerinden kaynaklanmaktadır. Bu zayıf yönler, güvenlik açıkları olarak da ortaya çıkabilir ki bu da donanım Truva atları gibi zararlı modifikasyonlara kapı açmaktadır [4, 5, 7–10].

ÇBÖT endüstrisi, üretim süreçlerini yönetmek için iki ana iş modeli (Şekil 1.2) kullanmaktadır [2]. Dikey model, tek bir şirketin tüm aşamaları kontrol etmesini içermektedir. Bu durum yüksek kontrol ve güvenlik sağlamakta, fakat maliyetleri artırmaktadır. Yatay model ise, tasarım, fabrikasyon, montaj gibi farklı süreçlerin çeşitli şirketler tarafından üstlenilmesini kapsamaktadır. Bu model maliyetleri düşürmekte ve esneklik sunmakta, ancak güvenlik ve tutarlılık kontrollerini zorlaştırmaktadır.

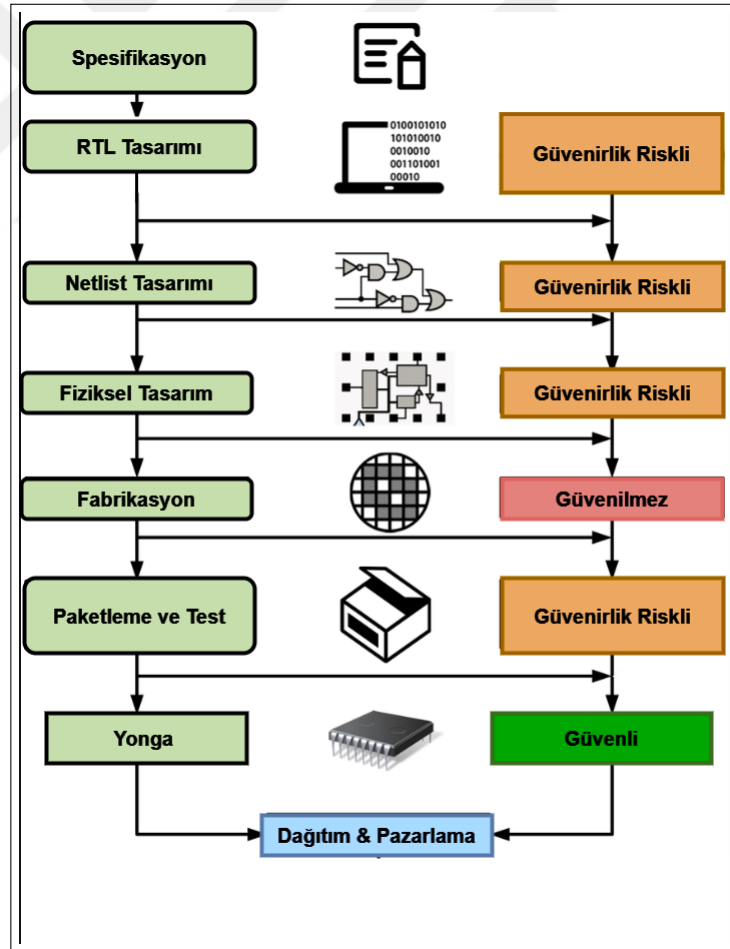


Şekil 1.3: Tümleşik Devrelerin Üretim Aşamalarının Firmalarla Birlikte Gösterimi.

Günümüzde ekonomik nedenlerden dolayı, çoğu yarı iletken şirketi bilgisayar yongaları üretmek için yeterli iç kaynaklara sahip değildir. Şekil 1.3 de üretim aşamalarının farklı şirketlerin elinde olduğu gösterilmiştir [3, 6]. Piyasada sıklıkla kullanılmakta olan ve yakından bilinen NVIDIA, AMD gibi firmaların fabrikaları yoktur. Tasarımlarının

bazı kısımlarını EDA araçlarını kullanarak kendi içlerinde yapmakta, bazı kısımlarını ise tümleşik devre üreticilerine (ing. IDM) hazırlatmaktadır. Tüm bu aşamaların sonrasında fabrikası olan TSMC gibi firmalar yonga üretimini gerçekleştirmektedir.

Fabrikaya sahip olan firma sayısı az olmasına rağmen, bu fabrikalar coğrafi ve ekonomik şartlar nedeniyle dünyada belirli ülkelerde bulunmaktadır. Şekil 1.3’da bulunan firmaların ülke dağılımları incelendiğinde, genellikle tasarım ABD ve Çin’de, fabrikasyon Güney Kore veya Tayvan’da, montaj ve test işlemleri Çin’de gerçekleşmektedir [6, 11]. Bu küresel yapı, maliyetleri düşürmekte ve pazarlara hızlı erişim sağlayarak avantajlar sunmaktadır, ancak güvenlik açıkları ve kalite kontrolü gibi çeşitli zorluklar yaratmaktadır. Artık bir zorunluluk haline gelen yatay model kullanarak birçok farklı firma ve ülkede üretim yapılmaktadır. Bu modeldeki en büyük sorun güvenliğin garanti altına alınmasıdır.



Şekil 1.4: Tümleşik Devrelerin Yaşam Döngüsü [3].

Donanım güvenilirliği kavramının popülerleşmesinin nedeni, askeri ve finansal sistemlerde kullanılan tümleşik devrelerin güvenilir olmalarının gerekliliği ve bu alanlarda donanımsal tehditlerin potansiyel varlıklarından kaynaklanmaktadır. Kritik sistemlere ait devreler, güvenilirliği garanti altına alınamayan fabrikalarda tasarlanmakta ve üretilmektedir. Üretilen bu devreler kritik sistemlerin güvenliğini doğrudan tehdit altında bırakmaktadır. Bu nedenle, devrelerin kullanıldığı kritik sistemler de güvenilir olarak tanımlanmaktadır [4, 5, 7–10].

Tümleşik devrelerin üretim süreçleri ve kullanılan farklı yöntemler, devrelerin güvenilirliği ve performansı üzerinde büyük bir etkiye sahiptir. Her bir süreç güvenlik tehditleri açısından farklı düzeylerde savunmasızlığa neden olmaktadır (Şekil 1.4). Aşağıda tümleşik devrelerin üretim süreci ve ilgili güvenlik riskleri detaylandırılmıştır:

- Spesifikasyon: Tümleşik devrenin fonksiyonel ve performans kriterlerinin tanımlandığı aşamadır. Bilgi sızıntısı dışında düşük güvenlik riskine sahiptir. Bu aşamada fiziksel ürün olmadığından dolayı fiziksel saldırılar mümkün değildir.
- RTL (ing. Register Transfer Level) Tasarımı: Devrenin davranışsal seviyede tasarlandığı aşamadır. Bu aşama, fonksiyonel bileşenlerin ve bağlantıların tanımlanmasını içermektedir. Üçüncü taraf fikri mülkiyetleri ve EDA araçları kullanıldığında güvenlik riskleri ortaya çıkabilir. Bu araçlar ve bileşenler, kötü niyetli kod veya arka kapılar içerebilmektedir.
- Netlist/Lojik Kapı Seviyesi: RTL tasarımın kapı seviyesine dönüştürüldüğü aşamadır. Bu da, devrenin mantıksal bileşenlere ayrılması anlamına gelmektedir. Güvenlik açısından, üçüncü taraf bileşenlerin entegrasyonu nedeniyle risk taşımaktadır.
- Fiziksel Tasarım ve Yerleşim (ing. Layout) Seviyesi: Netlist'in fiziksel düzlemdeki yonga üzerine yerleştirilmesini ve yolların çizilmesini içermektedir. Sarı renkle işaretlenmiş olması, bu aşamanın belirli savunmasızlıklara sahip olduğunu göstermektedir. Bu savunmasızlıklar, tasarımın optimize edilmesi sırasında ortaya çıkabilecek yan kanal saldırıları gibi risklerden kaynaklanabilmektedir.
- Fabrikasyon: Tümleşik devrenin fiziksel olarak üretildiği aşamadır. Bu, devrelerin gerçek silikon çipler üzerinde oluşturulmasını içermektedir. Yonga üretimi genellikle yüksek teknoloji gerektiren ve dış kaynak kullanımı yüksek olan

tesislerde gerçekleştirilmektedir. En yüksek güvenlik riskine sahip aşamadır. Üretim sürecindeki hatalar veya kötü niyetli değişiklikler, devrenin bütününe zarar verebilmekte ve güvenlik açıklarına yol açabilmektedir.

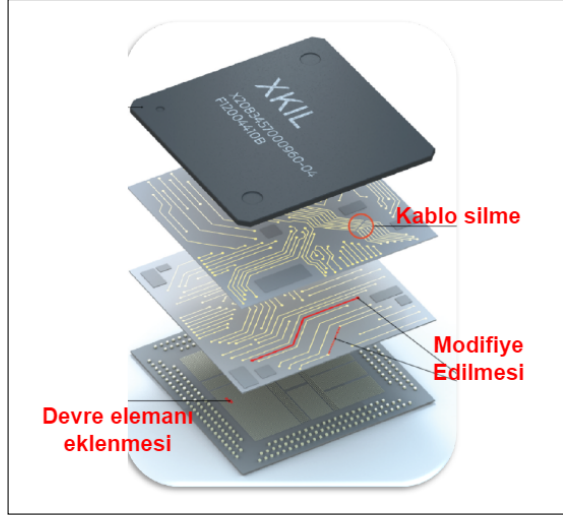
- Paketleme Aşaması: Üretilen yongaların birleştirilerek son ürün haline getirildiği aşamadır. Çiplerin paketlenmesi ve çeşitli bileşenlerin bir araya getirilmesini içermektedir. Paketleme sırasında eklenebilecek kötü niyetli bileşenler güvenlik açıklarına neden olmaktadır.
- Pazar: Üretilen tümleşik devrelerin nihai kullanıcıya ulaştırıldığı aşamadır. Bu aşama genellikle güvenilir kabul edilmekte, ancak ürünlerin yanlış dağıtımı veya sahteciliği gibi riskler içerebilmektedir.

Tümleşik devrelerin üretimindeki karmaşık yapıyı ve bu süreçler içerisinde ortaya çıkan güvenlik problemlerini kısaca inceledik. Her bir aşamanın farklı ve kendilerine özgü güvenlik zaafiyetleri içerebildiği, bu nedenle, etkin güvenlik protokolleri ve kontrol sistemlerinin gerekliliği açıkça ifade edildi.

1.2. Truva Atları

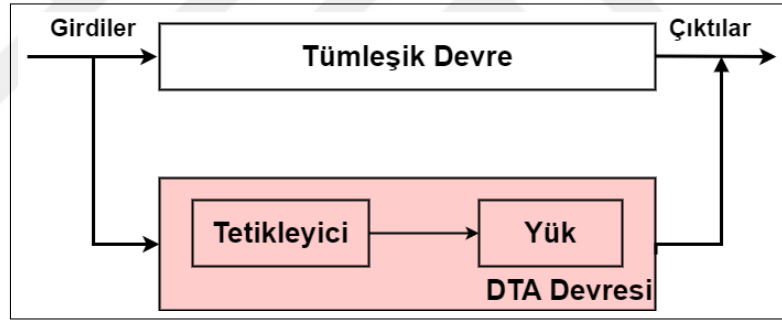
Donanımsal Truva Atı (DTA), bilgisayar sistemlerindeki donanım bileşenlerine kasıtlı olarak gizlice yerleştirilmekte olan kötü niyetli değişiklik olarak tanımlanmaktadır. Kastedilen değişiklikler Şekil 1.5 de gösterilen; ekstra devre elemanları eklemek veya mevcut elemanların modifiye edilmesi olmaktadır. Bu zararlı bileşenler, normal operasyon sırasında devrenin işleyişine hiçbir zarar vermemekte; ancak belirli gizli komutlarla veya tetikleyicilerle aktive edildiklerinde, istenmeyen işlevleri başlatarak cihazın işleyişini bozabilmekte veya güvenliğini tehlikeye atabilmektedirler. Truva atları genellikle güvenlik açıklarını istismar etmek, çeşitli bilgileri sızdırmak veya sistemin normal işleyişini bozmak amacıyla kullanılmaktadırlar. Bu tehlikeler, özellikle kritik altyapılarda veya yüksek güvenlik gerektiren sistemlerde ciddi riskler oluşturmaktadır [12].

Truva atları, üretim sürecinin herhangi bir aşamasında entegre edilebilmektedir. Özellikle üçüncü taraf bileşenlerin kullanımı ve tümleşik devrelerin fabrikasyon süreci gibi kontrolün zor olduğu aşamalar bu tür saldırılara açıktır. Son yıllarda küreselleşen üretim zincirleri, Truva atların potansiyel olarak daha da yaygınlaşmasına yol açmaktadır.



Şekil 1.5: Katmanlar Üzerinde Truva Atının Gösterimi [1] .

Truva atlarının, yazılım tabanlı saldırılardan farklı olarak, fiziksel donanım üzerinde yapılan değişiklikler oldukları için tespit edilmeleri daha zordur [6].



Şekil 1.6: Donanımsal Truva Atı Modelinin Genel Gösterimi.

1.2.1. Etkileri

Truva atları, genellikle iki ana bileşenden oluşmaktadır (Şekil 1.6); bunlar da bir tetikleyici (ing. trigger) ve bir yük (ing. payload) kısmıdır [13].

- **Tetikleyici Kısmı:** Truva atının ne zaman aktif hale geleceğini belirleyen koşulu ifade etmektedir. Bu koşul, spesifik bir sinyal düzeni, belirli bir giriş kombinasyonu, sıcaklık gibi fiziksel bir değişken veya belirli bir zamanlayıcı durumu gibi dijital bir olay olabilmektedir. Tetikleyici, Truva atının pasif durumdan aktif duruma geçişini kontrol etmekte ve genellikle yüksek derecede gizlilik

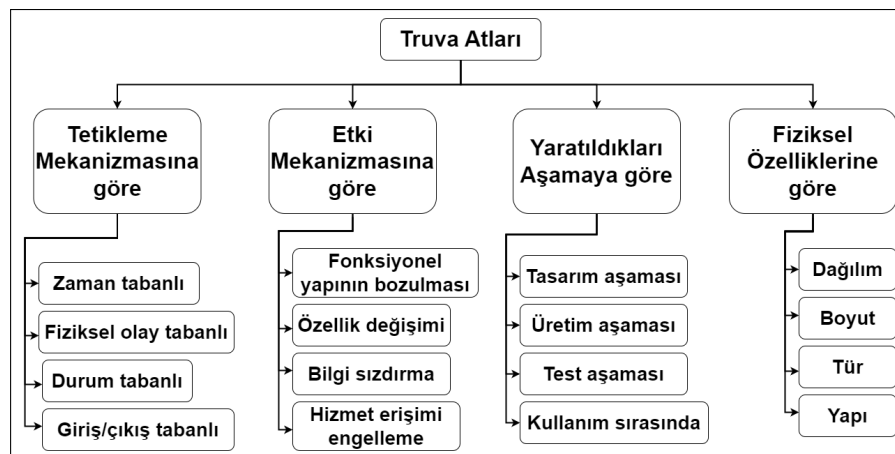
içermektedir.

- **Yük Kısmı:** Truva atının tetiklendiğinde gerçekleştirdiği veri sızdırma, işlevsel bozulma, yanıltıcı bilgi üretme veya sistemin tamamen devre dışı bırakılması gibi zararlı işlevleri ifade etmektedir. Yük, Truva atının tasarım amacına göre farklılık gösterir ve genellikle maksimum zarar veya istihbarat toplama kapasitesine odaklanmaktadır.

Truva atlarının zararlı etkileri ve güvenlik riskleri, ciddi boyutlarda güvenlik tehditleri oluşturmaktadır. Truva atları tarafından gerçekleştirilen gizlilik ihlalleri, hassas bilgilerin sızdırılması veya dışa aktarılması şeklinde meydana gelmektedir. Bu bilgiler, kullanıcıların kişisel verileri, şirket sırları veya ulusal güvenlikle ilgili kritik bilgiler olabilmektedir. Ayrıca, Truva atları cihazın normal işleyişini kesintiye uğratarak yanıltıcı sonuçlar üretmekte veya cihazın tamamen işlevsiz hale gelmesine neden olmaktadır. Bunun yanı sıra, Truva atları sistem güvenilirliğini azaltarak, özellikle askeri veya altyapı gibi kritik sistemlerin beklenmedik şekilde başarısız olmasına neden olmaktadır. Bu tür etkiler, Truva atlarının yaratabileceği risklerin ne kadar büyük ve ciddi olduğunu göstermektedir.

Tablo 1.1’de örnek bir senaryo üzerinde, bir Truva atının nasıl özelleştirildiği ve belirli bir tetikleyici tarafından nasıl aktive edildiği anlatılmaktadır [6].

1.2.2. Sınıflandırması



Şekil 1.7: Donanımsal Truva Atlarının Sınıflandırılması.

Tablo 1.1: Kritik Bir Sistemde Durum Tabanlı Truva Atı Örneği.

Senaryo:	Bir askeri uçakta kullanılan radar sistemini kontrol eden mikroçipte bir Truva atı bulunmaktadır. Truva atının tetikleyicisi, radar sisteminin belirli bir frekans bandında çalışmaya başlamasıyla aktive olmaktadır. Normal şartlar altında, radar sistemi bu frekans bandını sadece düşman uçakları tespit etmek için kullanmaktadır.
Tetikleme Mekanizması:	Radar sistemi belirli bir frekansa ulaştığında, mikroçip üzerindeki Truva atının tetikleyicisi bir iç devre kapısını aktive etmektedir. Bu tetikleyici, normalde pasif durumda olan ve sadece bu koşul altında devreye giren bir mantık kapısı kullanmaktadır (tetikleme: durum tabanlı).
Yük:	Tetikleyici aktive olduğunda, yük devreye girer ve radar sisteminin işlem algoritmasını değiştirir. Bu değişiklik, radar ekranında düşman uçaklarının yanlış konumlarını gösterir (etki mekanizması: fonksiyonelliğin değişmesi).
Sonuç:	Radar operatörü, düşman uçaklarının gerçekte olduğundan farklı bir konumda olduğunu görür. Bu durum, yanıltıcı bilgi nedeniyle savunma stratejilerinin yanlış yönlendirilmesine ve potansiyel olarak kritik hatalar yapılmasına yol açar.
Etki Analizi:	Bu tür bir Truva atı, askeri operasyonların güvenilirliğini ciddi şekilde tehlikeye atabilir. Yanıltıcı bilgiler, savunma kararlarını olumsuz etkileyebilir ve bu da stratejik hatalara ve gereksiz risklere neden olabilir.

Donanımsal Truva Atları, çeşitli sınıflandırma metriklerine göre ayrılmaktadırlar. Temel sınıflandırmalar; yaratıldıkları aşamaya , işlevselliklerine, tetikleme mekanizmalarına göre olmaktadır [12, 14]. Bu sınıflandırma çerçevesi, Truva atları daha iyi anlamamızı ve savunma stratejileri geliştirmemizi sağlamaktadır.

Ayrıca, tetiklendiğinde yükün sistemin işleyişine nasıl zarar verebileceğini de açıklamaktadır.

1. Tetikleme Mekanizmasına Göre Sınıflandırma: Truva atları aktivasyon koşullarına göre sınıflandırılmaktadır:

- Zaman Tabanlı Tetikleyiciler: Belirli bir zaman veya olaya dayalı olarak aktive olmaktadır. Örneğin, bir devre parçasının belirli bir tarih veya süre sonunda aktive olması.
- Fiziksel Olay Tabanlı Tetikleyiciler: Çevresel faktörlere duyarlı olan bu tetikleyiciler, sıcaklık, basınç veya ışık gibi fiziksel değişikliklere tepki vermektedirler.
- Durum Tabanlı Tetikleyiciler: Cihazın belirli bir duruma ulaşmasıyla aktive olmaktadır. Bu durum, belirli bir işlemci komutu, işlemci yükü veya bellek durumu olabilir.
- Giriş/Çıkış Tabanlı Tetikleyiciler: Belirli bir giriş veya çıkış işlemi gerçekleştiğinde aktive olmaktadır. Örneğin, bir ağ bağlantısı başladığında veya USB cihazı takıldığında devreye girmektedirler.

2. Etki Mekanizmasına Göre Sınıflandırma: Truva atları, gerçekleştirdikleri zararlı işlevlere göre sınıflandırılmaktadır:

- Devrenin Fonksiyonelliğinin Değişmesi: Devre çıktılarının beklenenden farklı olmasıdır. Devrenin fonksiyonel yapısı etkilendiği için yapması gereken işi gerçekleştirememiş ve beklenen çıktıyı üretememiştir.
- Devrenin Özelliklerinin Değişmesi: Devre hızının veya güç tüketimi değerlerinin olması gerekenden farklı çıkmasıdır. Bu etki de devrenin tasarlanırken sağlaması beklenen özelliklerinin değişmesine yol açar.
- Bilgi Sızdırması: Devre içindeki hassas, saklanan bilgilerin Truva atı aracılığıyla dışarı sızdırılmasıdır.

- Hizmet Erişimi Engellenmesi: Devredeki bazı kesmelerin (ing. interrupt) işleme alınmamasına neden olabilir.

3. Yaratıldıkları Aşamaya Göre Sınıflandırma: Truva atları üretim sürecinin hangi aşamasında yerleştirildiklerine göre de sınıflandırılmaktadır:

- Tasarım Aşamasında Yerleştirilenler: Mikroçip veya devre kartı tasarlanırken gizlice eklenmektedirler.
- Üretim Aşamasında Yerleştirilenler: Yonga fabrikasyon sürecinde, genellikle üçüncü parti kişi/kurumlar tarafından eklenmektedirler.
- Test Aşamasında Yerleştirilenler: Test cihazları veya arabirimler kullanılarak Truva atı devreye fiziksel olarak eklenebilir.
- Paketleme Aşamasında Yerleştirilenler: Montaj veya paketleme sırasında devreye ekstra kapılar veya modüller eklenebilir.

4. Fiziksel Özelliklere Göre Sınıflandırma: Truva atlarının hangi fiziksel metodlarla entegre edildiğini ve nasıl maskelendiğini açıklamaktadır:

- Dağılıma göre: Truva atının donanım üzerindeki konumu, tek bir bölgede mi yoksa yonga genelinde dağıtık olarak mı yer aldığına bakılır.
- Boyuta göre: Truva atının fiziksel boyutu, birkaç kapıdan başlayarak büyük bir yonga alanına kadar değişiklik gösterebilir. İçerdiği transistör/kapı sayısının yongaya oranına bakılarak, küçük veya büyük boyutlu olarak adlandırılır.
- Türe göre: Yeni bileşen eklendiyse fonksiyonel, mevcut bileşenin özellikleri değiştirildiyse parametrik olarak adlandırılır.
- Yapıya göre: Trojan'ın donanıma yerleştirilme şeklinin yerleşim planını değiştirip değiştirmemesine bakılır.

Tetikleyici ve yük ilişkisinin anlaşılması, Truva atı saldırılarını önlemede kritik bir rol oynamaktadır. Güvenlik sistemlerinin tasarımında, potansiyel tetikleyicilerin ve yüklerin çeşitliliği göz önünde bulundurularak, bu tehditlere özel savunma stratejileri geliştirilmelidir. Örneğin, çoklu tetikleyicilere dayanan saldırıları tespit etmek için daha kapsamlı izleme ve analiz sistemleri gerekmektedir. Aynı şekilde, karmaşık yük senaryolarına karşı korunmak için, sistemlerin hem veri sızıntılarını hem de işlev

bozukluklarını tespit edebilecek şekilde güçlendirilmesi önemlidir.

1.3. Problem Tanımı ve Kapsamı

Literatürde bulunan mevcut gecikme tabanlı tespit yöntemleri, artan miktarda varyasyon değişimlerinden, özellikle yonga içi değişim bileşeninden, küçük boyutlu Truva Atlarını tespit etme zorluğundan, çok küçük devrelerde doğrulama yapmaktan ve altın çip gereksinimini sorunlarından bir veya daha fazlasından muzdariptirler [12, 13, 15].

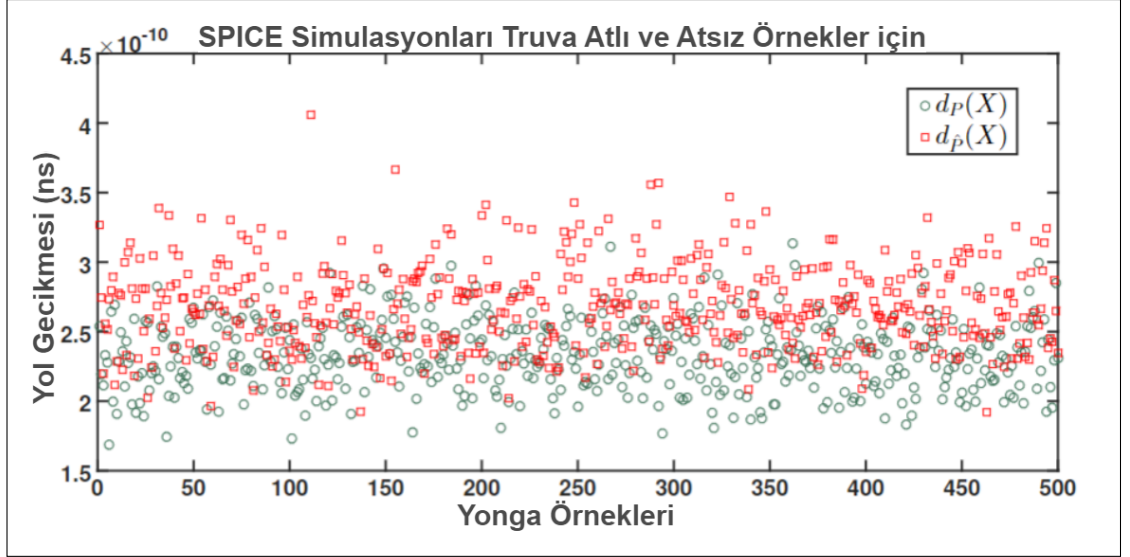
Bu tez çalışması literatürde konumsal korelasyonları (ing. spatial correlation) kullanarak Truva atını tespit eden ilk yaklaşımdır. Bunun içinde hem süreç varyasyonlarının kapsamlı bir modeli kullanılmış, hem de SPICE aracıyla aynı doğruluğa sahip hassas bir gecikme ölçüm yöntemi kullanılmıştır. Literatürde bulunan çalışmalarda [16–20] süreç varyasyonları modellenirken, rastgele parametreler kullanılmaktadır. Entegre devrelerde doğal olarak var olan ve varyasyon modeline önemli etkisi olan konumsal korelasyonlar bizim çalışmamıza kadar göz ardı edilmiştir.

Bu tez çalışmasında, varyasyonların etkisini konumsal korelasyondan faydalananak ortadan kaldıran, bu sayede Truva atının tespit edilmesini sağlayan gecikme tabanlı tespit yöntemi önerilmektedir. Yöntem, devrenin konumsal özelliklerini kullanarak, varyasyonların gecikme değerine olan etkisini azaltmakta ve Truva atlarının neden olduğu gecikme artışlarını daha belirgin hale getirmektedir.

Yöntemimiz, orjinal tasarımı değiştirmeyen ve zarara uğratmayan, yıkıcı olmayan yöntem sınıfındadır. Gerçekçi süreç varyasyonuna ait parametre değişimleri altında yalnızca bir XOR kapısından oluşan en küçük Truva atını bile tespit edebilmektedir. Ayrıca, tüm deneylerde pasif bir Truva atı varsayılsa bile, Truva atının etkinleştirilmesine karşı duyarsızdır.

Yöntemimizdeki ana fikir, birbirine yakın konumlarda kapılara sahip yol çiftlerini bulmaya dayanmaktadır. Böylece bu kapıların gecikme değerleri birbirlerine göre daha fazla korelasyona sahip olmaktadır. Ancak, yüksek derecede korelasyona sahip olan yol çiftlerini bulmak oldukça zor bir görevdir ve doğru şekilde ele alınmalıdır. Tez çalışması boyunca yol çiftleri bulmanın hem zaman hem de hafıza kullanıma göre verimliliği üzerinde yapılan çalışmalar gösterilmiştir.

Öncelikle, sadece gecikme değerlerini ölçerek truva atı tespit etmenin neden mümkün ol-



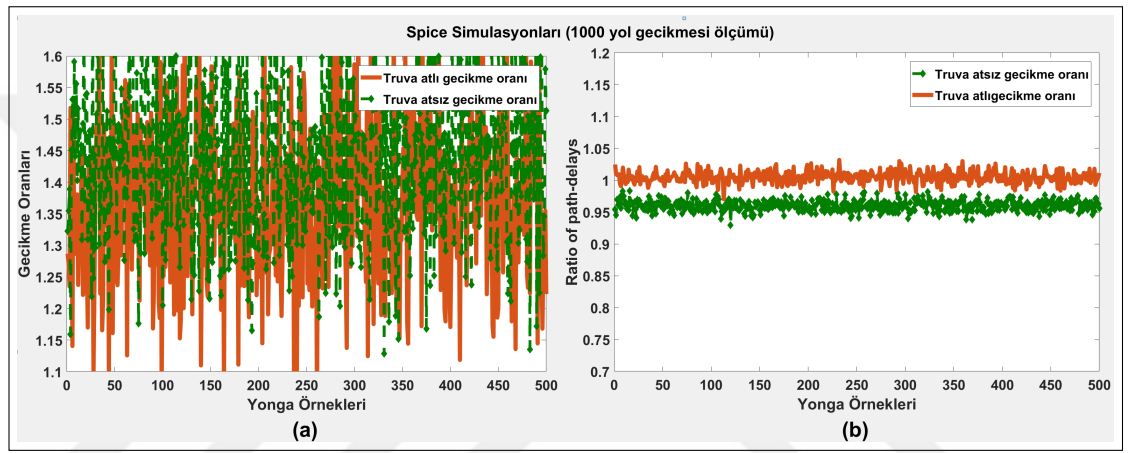
Şekil 1.8: c7552 devresindeki bir yola ait truva atlı ve atsız gecikme değerleri.

madığını inceleyelim. Burada varyasyonun gecikme üzerindeki etkisini açıkça anlamış olacağız. Aynı zamanda literatürde gecikme temelli truva atı tespiti konusunda en çok referans gösterilen ve karşılaştırma yapılan çalışmalar olan [21] ve [17] ile de kıyaslamış olacağız. Çalışmalar [17, 21] da sadece yol gecikmesi değeri incelenerek Truva atı tespiti yapılmaktadır. Gecikme tabanlı Truva atı tespiti için en bilinen yöntemlerden biri olan [21] çalışması, yol gecikmesinin parmak izini kullanarak Truva atı eklenmiş örnekleri Truva atı içermeyenlerden ayırt etmektedir. Yöntem, Truva atının neden olduğu ek gecikme bileşenini ortaya çıkarmayı amaçlamaktadır. İlk olarak, Truva atı içermeyen örneklerin, yani altın çiplerin gecikme parmak izi toplanmakta, ardından test çipleri, eldeki altın çip sonuçları ile karşılaştırılarak sınıflandırılmaktadır. Bu çalışmalardan sonra "Kendi varyasyon modelimiz ve yol gecikme ölçüm sonuçlarımızı kullanarak, doğrudan küçük boyutlu Truva atının etkisini gözlemleyebilir miyiz?" sorusunu cevapladık.

Bu nedenle, Bölüm 1.4'de detaylı açıklanan varyasyon modelimize göre 1000 yonga örneği ürettik. Yalnızca altı adet 2 girişli lojik kapılardan oluşan kısa bir yol seçtik [17]. Elimizdeki 1000 örneğin yarısı olan 500 tanesine Truva atı ekledik. Bunun için, yükü Şekil 3.5'de gösterildiği şekilde ekledik. Her bir örnek için transistor seviyesi SPICE simülasyonu yaptık. Truva atı içermeyen ($d_P(X)$) ve Truva atı eklenmiş ($d_{\hat{P}}(X)$) örnekler için ortaya çıkan gecikmeler Şekil 1.8'te gösterilmektedir. Şekil 1.8'de x-ekseni farklı yonga örneklerini, y-ekseni ise gecikme değerlerini göstermektedir.

Şekil 1.8, kısa bir yol için bile, ortaya çıkan Truva atı gecikmesinin varyasyonların

altında gizlendiğini açıkça göstermektedir. Truva atı içermeyen ve Truva atı eklenmiş örnekler o kadar iç içe geçmiştir ki, iki sınıfı ayırt etmek için bir eşik bulunmamaktadır. Bu iki yöntemin [17, 21] bir arada kullanılması bile, ISCAS'85 devreleri üzerinde küçük boyutlu olarak nitelendirilen bir adet XOR kapısına sahip truva atını tespit etmeyi başaramamıştır. Varyasyon etkisi, Truva atının gecikmeye olan etkisinden oldukça fazla olduğu için tespiti zorlaştırmaktadır. Bu nedenle, yol gecikmesi kullanarak küçük boyutlu truva atı tespiti yapabilen gelişmiş yöntemlere ihtiyaç duyulmaktadır. Tezin literatüre yaptığı en önemli katkı bu şekilde özetlenebilir.



Şekil 1.9: Yol çiftlerinin gecikme oranları (a) rastgele konumlarda bulunan, (b) birbirlerine yakın konumlarda bulunan

Şekil 1.9'da ISCAS'85 [22] c7552 32 bit toplayıcı/karşılaştırma devresinden seçilen iki adet yola ait gecikme değerlerinin oranları karşılaştırılmaktadır. Her iki şekilde de Truva atı olarak Şekil 3.4'deki XOR kapısı, şüphelenilen yolda ilgili kenara eklenmiştir, örneklerin yarısı (yeşil) Truva atı içermeyenken, diğer yarısı (kırmızı) Truva atı içermektedir. Tüm gecikmeler, transistor seviyesi SPICE [23] simülasyonlarına göre hesaplanmıştır.

Şekil 1.9, hem varyasyonun etkisini net ortaya koymakta hem de yöntemimizin varyasyonun etkisinden kurtulmak için sağladığı başarıyı göstermektedir. Şekil 1.9 (a)'da, Şekil 1.8 ile tamamen aynı şüpheli yol kullanılmıştır. Seçilen bu şüpheli yol ile ilişkili olması içinde, rastgele başka bir yol seçilmiştir. Şekil 1.9 (b)'de tekrar Şekil xx'deki şüpheli yol kullanılmıştır, bu yol ile yakın konumlardan geçen başka bir yol ilişkili yol

olarak seçilmiştir. Şekil 1.9 (a) ve (b) de x-ekseni farklı yonga örneklerini, y-ekseni ise gecikme oran değerlerini göstermektedir.

Şekil 1.9 (a) yöntemimize neden ihtiyaç duyulduğunu açıkça göstermek için oluşturulmuştur. Rastgele konumlarda bulunan iki yolun gecikme oranları göstermektedir. Şekildeki gecikme oranları konumsal korelasyonlar dikkate alınmadan çıkarılan yollar olduğu için, oran değerleri birbiriyle iç içe geçmiş yani Truva atı tespit edilemez haldedir.

Şekil 1.9 (b) ise yöntemimizin sağladığı başarıyı ortaya koymak için seçilmiş herhangi bir test örneğidir. Kapıların birbirine yakın konumlarda bulunduğu iki yolun gecikme oranını göstermektedir. Sonuçta elde edilen oran değerleri, istendiği gibi sabit bir değer aralığındadır. Böylece birbirine yakın konumlarda olan yolların kullanılmasıyla varyasyonun etkisinin ortadan kaldırıp, küçük boyutlu Truva atının bile başarıyla tespit edilebileceği gösterilmiştir.

1.3.1. Problemin Önemi

DTA konusuna ait ilk uygulamalar, ABD ile Rusya arasındaki soğuk savaş dönemine dayanmaktadır. Bu dönemde Rusya, ABD'ye gidecek olan daktiloları ele geçirip, üzerlerine anahtar kaydedici yapılar eklemiştir [24].

Donanım bazında tespit edilen ilk arka kapı, 2012 yılında Actel FPGA üzerinde bulunmuştur [25]. Yakın zamanda Avusturya güvenlik kurumu SEC Consult, Barracuda firmasına ait birçok üründe raporlanmamış arka kapılar bulunduğunu açıklamıştır. Firma, bunların üretim aşamasındaki deneysel çalışmalarda kullanıldığını ve sonradan unutulduğunu belirtmiştir. Bu, dolaylı da olsa arka kapıların varlığını kabul etmektedir [26].

Bir diğer örnekte, Amerika Ulusal Güvenlik Ajansı (NSA), Huawei ve ZTE gibi Çin merkezli firmaların haberleşme cihazlarında Truva atı içerdiğine yönelik raporlar yayınlamıştır. Bu raporlar sonucunda, bu cihazların kritik projelerde kullanılması yasaklanmıştır [27]. Bu tür cihazların Truva atı içermesi, ulusal güvenlik açısından ciddi tehditler oluşturabilir ve kritik altyapıların güvenliğini riske atabilir.

Endüstriyel alanda, Truva atlarının etkisi ve tespiti üzerine birçok haber ve rapor yayınlanmıştır. Örneğin, Bloomberg Businessweek'in 2018 yılında yayınladığı bir makalede [28], Supermicro'nun anakartlarında Çinli ajanlar tarafından yerleştirilen mikroçipler

hakkında iddialar yer almıştır. Bu iddialar, donanım güvenliği konusundaki endişeleri artırmış ve küresel tedarik zincirlerinde güvenliğin önemini bir kez daha gözler önüne sermiştir.

2020 yılında Xilinx'in FPGA'larında tespit edilen StarBleed güvenlik açığı, donanım trojanlarının tespitine yönelik yeni bir örnek olarak öne çıkmıştır. Araştırmacılar, bu güvenlik açığını kullanarak FPGA'ların bitstream dosyasını deşifre edebilmiş ve bu şekilde donanımsal truva atı yerleştirme potansiyelini göstermiştir [29].

Güncel konferans ve dergi içeriklerinin kategorilerini incelediğimizde, donanım güvenliği başlığı artık siber güvenlik başlığından ayrı tutulmaktadır. Bu da donanım güvenliği başlığı altında bulunan DTA tespitinin alt başlık olarak kabul edildiğinin göstergesidir.

Bu bağlamda, Spectre ve Meltdown [30] gibi modern işlemcilerde tespit edilen güvenlik açıkları, donanım güvenliğinin önemini bir kez daha gözler önüne sermiştir. 2018 yılında keşfedilen bu açıklar, işlemcilerin spekülasyon yürütme mekanizmalarından yararlanarak bellekten izinsiz veri sızıntısına olanak sağlamıştır. Bu tür güvenlik zafiyetleri, yalnızca yazılım değil, aynı zamanda donanım bileşenlerinin de kritik öneme sahip olduğunu ortaya koymuştur. Bu olaylar, donanım tabanlı tehditlerin yalnızca teori değil, pratikte de gerçekleşebilir olduğunu göstermiş ve donanım güvenliği alanında yeni araştırmaların yapılmasına zemin hazırlamıştır.

1.3.2. Yaklaşımımız

- Devrelerin üretimi sırasında, tüm devrelere değil sadece bazılarına Truva atı eklenmektedir. Bunun nedeni de Truva atı tespitini zorlaştırmaktır. Çünkü üretimden gelen tüm devrelerde Truva atı bulunması, hem tespit edilme olasılığını ciddi oranda arttırmakta hem de Truva atının doğası gereği gizli kalmaya ters düşmektedir.

Yöntemimiz, test devrelerinin yarısında Truva atı olduğunu varsaymaktadır. Literatürde bulunan Truva atı tespit yöntemlerinden [20, 21, 31–34], altın devreye ihtiyaç duymayan yan-kanal analizi yöntemleri de aynı varsayıma sahiptirler.

- Altın devre ihtiyacı, Truva atı tespit yöntemlerinden yan-kanal analizi yöntemlerinde bulunmaktadır. Altın devre, içinde Truva atı olmadığından yani devrenin temiz olduğundan emin olunan devredir. Altın devre, Truva atı tespitinde

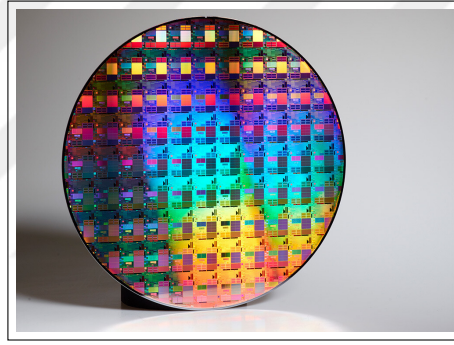
kıyaslama referansı olarak kullanılmaktadır. Yöntemimiz, gerekli referans modeli olarak altın devre kullanmadan Truva atını tespit edebilmektedir. Böylece, gecikme yöntemlerinin dezavantajlarından biri olan altın devreye olan ihtiyacı ortadan kaldırmaktadır.

- Gölge yazmacı (ing. shadow register) [35, 36] ve halka osilatörü (ing. ring oscillator) [37] kullanan bazı gecikme tabanlı Truva atı tespit yöntemleri, devrede tasarım değişikliğine ihtiyaç duymaktadır. Yöntemimiz yonganın tasarımına ilave eleman ekleme gibi değişikliklere ihtiyaç duymaz. Devreye zarar vermediği gibi spesifikasyonlarını da etkilememektedir.
- Yonga tasarımını gerçekleştirdiğimizi ve gerçek yerleşim planının elimizde olduğunu varsayıyoruz. Tasarımımızın üretime gittiğini ve üretim tamamlandıktan sonra tüm yol testlerini gerçekleştirdiğimizi varsayıyoruz.
- Güvenilmeyen tek kısmın fabrikasyon aşaması olduğunu varsayıyoruz ve sonrasında yongalar üzerinde yine kendi testlerimizi gerçekleştiriyoruz.
- Truva atlarının yük kısmı orijinal ve temiz olan yongaya bir miktar gecikme eklentisi getirir. Varsayımımız, eklenen Truva atlarının yol gecikmesi ölçümünü etkilemesi gerekliliğidir. Bazı Truva atları yol gecikmesi ölçümlerini neredeyse hiç etkilemeyebilirler. Bu tür Truva atları bizim yöntemlerimiz kullanılarak tespit edilemezler.
- Literatürdeki yöntemlerin çoğunda kullanılan test devrelerinde [38], her bir devreye bir adet Truva atı eklenmektedir. Ve bu truva atının devreyi etkilediği kısıtlı alanlar şüpheli olmaktadır. Yöntemimizde devredeki herhangi bir kenarının Truva atı devresi yerleştirme potansiyeline sahip olduğunu varsayıyoruz. Böylece devredeki her kenar şüpheli olmaktadır. Aslında bu varsayımımız, test şartlarını oldukça zorlaştırmaktadır. Literatürdeki birçok yöntemin aksine yöntemimiz, ilgili test devresinin her bir kenarı üzerinde truva atı tespiti yapabilme yeteneğine sahiptir.
- Devrenin birincil girişleri ve birincil çıkışlarına doğrudan eklenmiş Truva atı, devrenin spesifikasyon testleri sırasında kolaylıkla tespit edilebilmektedir [12, 16]. Bu nedenle, devredeki tüm kenarların şüpheli kenar olarak kabul edilmesine ek olarak, devrenin giriş/çıkışlarına Truva atı eklenmesinin gerçekçi olmadığını

varsayıyoruz. Devrenin giriş/çıkışlarına doğrudan bağlı kenarları şüpheli olarak kabul etmiyoruz.

1.4. Süreç Varyasyonu

Entegre devre üretim süreci, her biri yonga olarak adlandırılan bir dizi birbirinin aynısı olan devreyi tek bir yonga plakası (ing. wafer) üzerinde gruplandırır. Üretim süreci sona erdikten sonra, yonga plakası üzerindeki her bir yonga, eğer tüm testleri geçebilirse ayrı ayrı paketlenir, aksi takdirde ıskartaya çıkarılır. Testler, yonganın tasarımda belirlenen hız gereksinimlerini karşılayıp karşılamadığına karar vermek için zamanlama doğrulamasına tabi tutulur. Şekil 1.10, TSMC 100nm yongasının [39] üretimi için kullanılan levhalardan birini göstermektedir.

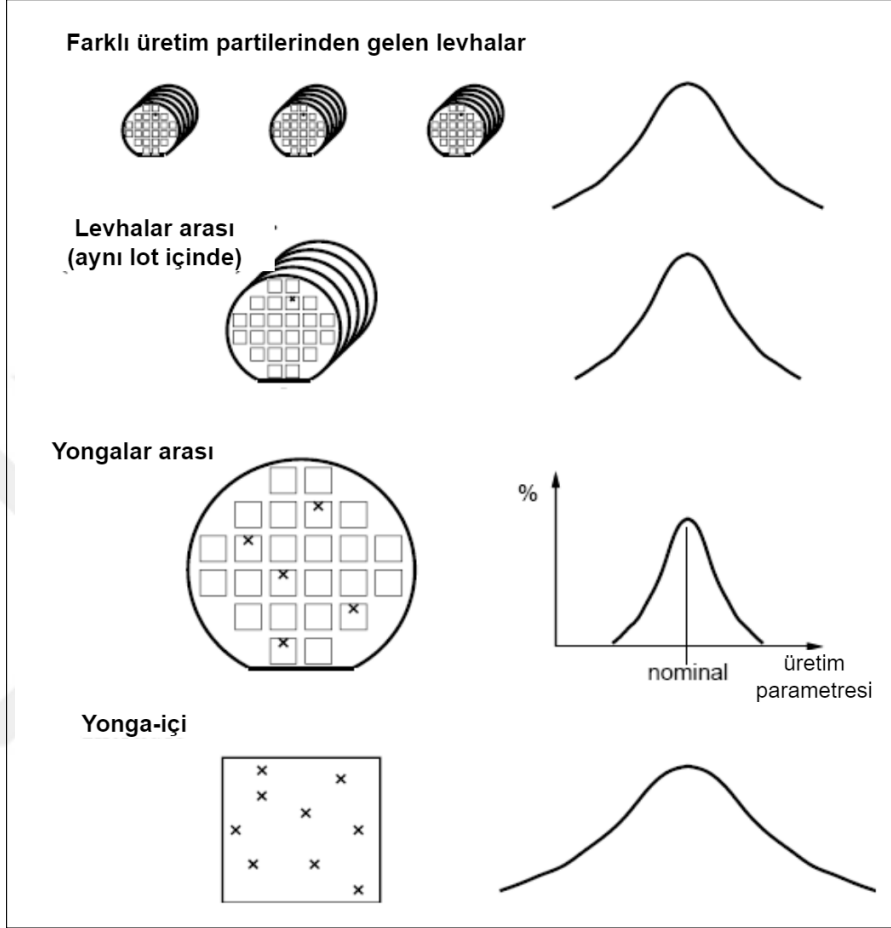


Şekil 1.10: Levha örneği: TSMC 100nm yonga üretimi.

Süreç varyasyonu, yonga üretim sürecinin doğasından kaynaklanmaktadır. Üretim sırasında devrenin fiziksel parametrelerindeki sapmaları ifade eder. Modern tümleşik devre üretim sürecinde asla istenmeyen ama ihmal edilemez olan bir bileşendir. Ekipman değişiklikleri, çevresel koşullar, malzeme özellikleri ve insan hataları gibi faktörlerden kaynaklanabilirler.

Devreler fonksiyonellik, hız ve güç tüketimi gibi belirli kriterlere göre tasarlanırlar. Üretim sonrası aşamada ise, üretimle elde edilen yonga seti bu kriterleri karşılayıp karşılamadığını görmek için incelenir. Ancak, kapı uzunluğu (ing. gate length), oksit kalınlığı (ing. oxide thickness) ve eşik voltajı (ing. threshold voltage) gibi çeşitli devre bileşenleri üzerinde, üretim sürecindeki varyasyonlar nedeniyle oluşan farklılaşmalar, yongaların gereksinim özelliklerinde de sapmalara neden olmaktadır [40, 41]. Bu

nedenle tamamen aynı gereksinim özelliklerine sahip olması için tasarlanıp, üretime gönderilen yongalar, üretim sonrasında tam olarak aynı gereksinimleri karşılayamamaktadırlar. Bunun yerine üretilen her bir yonga farklı özelliklere sahip olmaktadır.

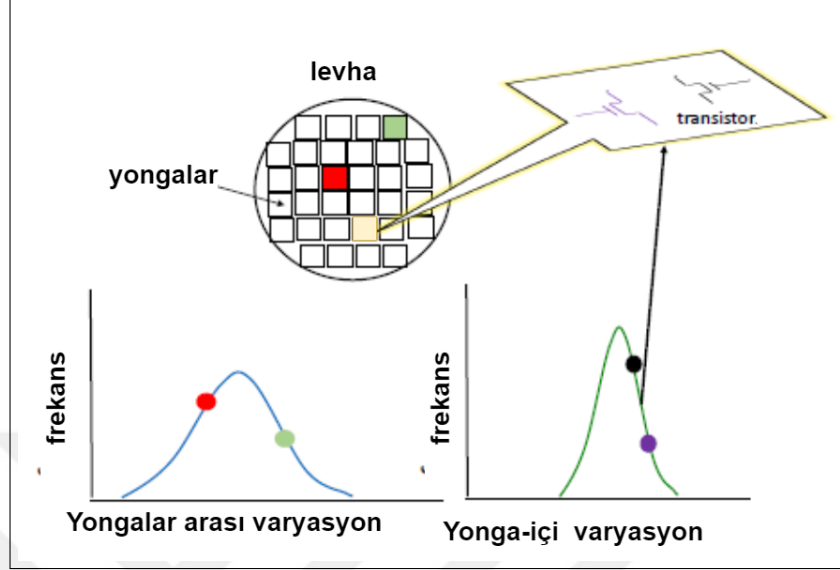


Şekil 1.11: Süreç varyasyonlarının sınıflandırılması.

Süreç varyasyonları dört ana başlık altında adlandırılmaktadırlar [42]. Bunlar Şekil 1.11’da gösterilmektedir.

- Farklı üretim partileri (ing. lot-to-lot) arasında: farklı üretim partileri arasında meydana gelen varyasyonlardır. Doğaları gereği rastgele etki ederler.
- Farklı levhalar (ing. wafer-to-wafer) arasında: aynı üretim partisinde yer alan farklı levhalar arasında görülen varyasyonlardır.
- Farklı yongalar (ing. inter-die) arasında: aynı levha üzerindeki farklı bölgeler arasında meydana gelen varyasyonlardır.

- Yonga-içi (ing. intra-die): aynı yonga üzerindeki farklı bölümler arasında görülen varyasyonlardır. Konumsal korelasyondan etkilenirler.



Şekil 1.12: Yongalar-arası ve yonga-içi değişimlerin frekans parametresi üzerinde gösterimi.

Varyasyonların parametre değişimleri, konumsal özelliklerine göre iki ana başlık altında incelenmektedir [43, 44].

1. Yongalar Arası (ing. inter-die) Varyasyonlar: Parametreler, aynı levha üzerindeki iki farklı yonga arasında veya işlenen levha yenisiyle değiştirildiğinde veya yeni bir levha işlenmeye başladığında değişebilir. Bu varyasyon, levhaların hizalanması ve maske modellerine ait ekipmanın kalibrasyonundaki önlenemez değişikliklerin bir sonucudur. Farklı levhalar veya levha lotları arasındaki işlem parametrelerinin değişimini de kalıplar arası varyasyonlar altında incelenebilmektedir. Aynı levha üzerindeki tüm birimler yongalar arası değişimlerden birebir aynı şekilde etkilenir. Örneğin, kapı uzunluğunun (L) yongalar arası değişime sahip rastgele bir parametre olduğunu varsayalım. Eğer sadece yongalar arası değişimler varsa, aynı yongadaki tüm transistörler aynı L değerine sahip olacaktır. Yongalar arası varyasyon yalnızca farklı yongalardaki transistörlerin farklı L değerlerine sahip olmasına neden olmaktadır.
2. Konumsal Korelasyon ile Yonga-içi Varyasyonlar (ing. intra-die): Parametre

varyasyonları aynı yongadaki cihazlar için bile mevcut olabilir. Bu varyasyonlar yonga-içi varyasyonlar olarak adlandırılır. Örneğin, yonga-içi varyasyonlar nedeniyle, bir yongadaki bir transistör, her ikisi de aynı kapı uzunluğuna sahip olacak şekilde tasarlanmış olmasına rağmen, aynı yongadaki başka bir transistörden farklı bir kapı uzunluğuna (L) sahip olabilir. 20 yıl önce, yonga-içi varyasyonlar toplam varyasyonların ihmal edilebilir bir bölümünü oluşturuyordu. Ancak teknolojinin 90nm seviyesinin altına inmesiyle birlikte, yonga-içi varyasyonların toplam varyasyona olan yüzde katkısı %50'ye yaklaşmıştır [40, 41]. Örneğin, teknoloji 250nm'den 70nm'ye geçerken L kalıp içi varyasyonların yüzdesi %40'tan %65'e yükselmiştir [40]. Günümüzde kullanılan 5nm teknolojisinde ise L kalıp içi varyasyonların yüzdesi, yeni üretim ve malzeme teknolojileri kullanılarak %50 seviyelerinde sınırlandırılmıştır [41, 45]. Yonga içi varyasyonlar konumsal korelasyona sahiptirler, yani bir transistör parametresinin korelasyonu o cihazın konumuna bağlıdır, bunun nedeni yonga-içi varyasyonlara neden olan üretim süreçlerinin çoğunun bir konumdan diğerine kademeli olarak değişmesidir. Konumsal korelasyon nedeniyle, birbirine yakın cihazlar benzer şekilde etkilenirken, birbirinden uzak cihazlar daha ilişkisiz parametrelere sahip olmaktadır. Başka bir deyişle, aynı yonga üzerindeki iki birimin karşılık gelen parametrelerinin korelasyonu, aralarındaki mesafe azaldıkça artar. Literatürde konumsal korelasyonları gerçekleştirmek için farklı modeller bulunmaktadır.

Her geçen yıl transistör boyutlarının küçülmesi nedeniyle, süreç varyasyonunun devre parametreleri üzerinde neden olduğu sapma miktarları da artmaktadır. Dolayısıyla, devrenin yan kanal bilgilerinde gözlenen varyasyon etkileri gittikçe artmaktadır. Bu nedenle Truva atının etkisinin varyasyon etkisinden ayırt edilmesi daha zor hale gelmekte, bunun sonucunda Truva atı tespiti oldukça zorlaşmaktadır. Bu durum, adil bir Truva atı tespiti için gerçek devre özelliklerini temsil eden süreç varyasyonlarının modellenmesi gerektiğini göstermektedir. Varyasyonları modellemek için istatistiksel modellerden yararlanmaktayız.

İstatistiksel tasarım, olasılıksal modeller ve yöntemler kullanarak devre tasarımının süreç varyasyonları altında analiz etme ve optimize etme yöntemidir. Örneğin, Monte Carlo simülasyonları [13], istatistiksel statik zamanlama analizi [44] veya istatistiksel optimizasyon teknikleri kullanarak devre performansının olasılık dağılımı ve hassasiyeti

tahmin edilebilmektedir. Bu çalışmada istatistiksel zamanlama modellerinden [44, 46] kullanılarak, devredeki süreç varyasyonları gerçeğe en yakın şekilde modellenmektedir. Varyasyon modeline dair detaylar Bölüm 3.2’de anlatılmaktadır.

1.5. Tezin Organizasyonu

Bu bölüm tezde yer alan bölümlerin ve içeriklerinin genel bir özetini içermektedir. Her bir bölüm, belirli bir konuyu ele almakta ve bu konuda yapılan çalışmaları detaylandırmaktadır.

Birinci Bölümde, tümleşik devrelerin genel tanımı ve önemi, Truva atları ve etkileri, problemin tanımı ve kapsamı ele alınmıştır. Ayrıca, araştırmamızın motivasyon kaynakları ve yaklaşımımız hakkında bilgiler verilmiştir.

İkinci Bölüm Literatürde yer alan Truva atı tespit yöntemleri ve bu yöntemlerin sınıflandırılması incelenmiştir. Tersine mühendislik, mantıksal test, yan kanal analizi ve yol gecikmesi yöntemleri detaylandırılmıştır.

Üçüncü Bölüm Yönteme Hazırlık kısmıdır. Bu bölümde, devre çizgesi, Truva atı modeli, varyasyon modelimiz, gecikme ölçümü için kullanılan stokastik lojik efor modeli hakkında bilgilere yer verilmiştir. Ayrıca, yöntemlerde kullanılan yol gecikme ölçümlerinin doğrulanması ele alınmıştır.

Dördüncü Bölüm, Konumsal Korelasyonlardan Yararlanarak Yol Gecikmesi ile Truva Atı Tespitidir. Tezde kullanılan yöntemlerin tamamını içermektedir. Yöntemlere ait detaylar aşamalar halinde açıklanmıştır.

Beşinci Bölüm, geliştirilen yöntemlere ait deneysel sonuçları ve çalışma zaman analizlerini içermektedir. Korelasyon katsayısına dayalı Truva tespit yöntemi, ortak ızgaraların oranına göre Truva atı tespit yöntemi ve rastgele uyarlanabilir yöntem ile Truva atı tespiti sonuçları verilmiştir. Ayrıca, Truva atının tetikleyici kısmının tespit edilebilirliği incelenmiştir.

Altıncı Bölümde, FPGA Uygulamalarına dair tüm detaylar adım adım anlatılmaktadır. Bu bölümde, FPGA üzerinde yol gecikme ölçümü için tasarlanan yapı, süreç varyasyonunun etkisi ve Truva atının etkisi konuları ele alınmıştır. Ayrıca, Truva atının boyut değişimi incelenmiştir.

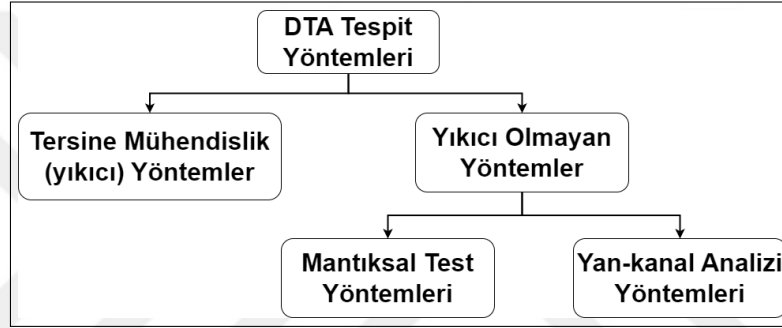
Son Bölümde, tez çalışmasının sonuçlarına dayalı olarak gelecekte yapılabilecek çalışmalar ve araştırma yönleri tartışılmıştır.



2. Literatür Özeti

2.1. İlgili Çalışmalar: DTA Tespiti Yöntemleri

Bu bölümde, Truva atı tespit yöntemleri genel bilinen (Şekil 2.1) başlıklarına göre ayrılarak incelenmektedir. Her bir başlık altında, o alanda en çok atıf alan, saygın yerlerde yayınlanmış ve yayım tarihi yeni olan yayınlar kullanılarak konular özetlenmektedir. Önemli alt başlıkların sonunda, anlatımda kullanılan kaynakların karşılaştırmalı tablosu özet olarak eklenmiştir.



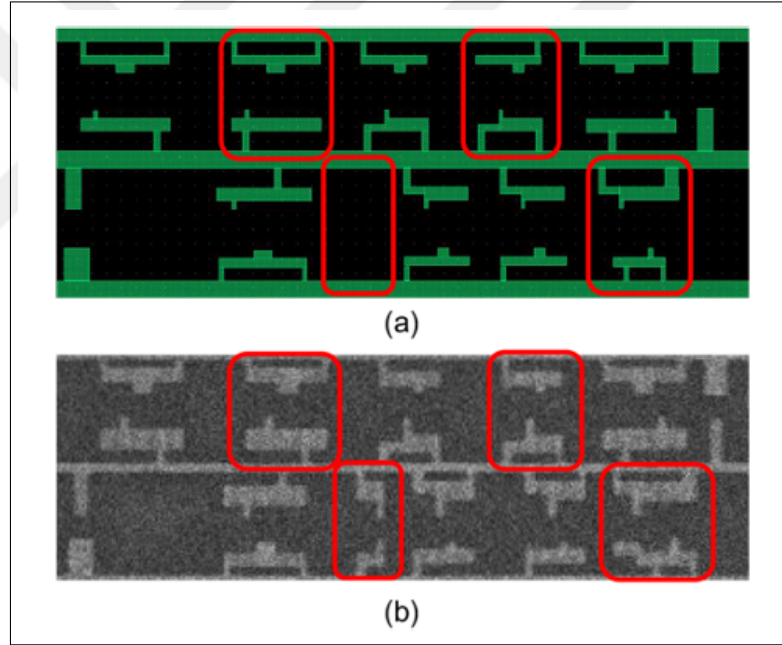
Şekil 2.1: Truva atı tespit yöntemlerinin sınıflandırılması.

2.1.1. Tersine Mühendislik Yöntemleri

Tersine mühendislik (ing. reverse engineering), devrelerin fiziksel olarak incelenmesini içerir ve bu süreç genellikle çipin katman katman soyulmasını ve her katmanın elektron mikroskopuyla incelenmesini gerektirir. Tersine mühendislik yöntemlerinden yıkıcı (ing. destructive) yöntemler olarakta bahsedilmektedir. Bu yöntem, kullanılan cihazlar ve gereken uzman mühendis ihtiyacı nedeniyle oldukça maliyetli ve zaman alıcıdır. Devre çeşitli kimyasallarla katmanlarına ayrıştırıldığı için işlem sonunda elde edilen devre de kaybedilmiş olur. Bunlar yöntemin dezavantajlarıdır. Son yıllarda makine öğrenmesi ve görüntü işleme yöntemlerinde görülen hızla ilerleme tersine mühendislik alanına da etki etmiştir. Çalışma [47]'den alınan fotoğrafta, Şekil 2.2 (a) 'da Tasarım aşamasından yerleşime ait görüntü, Şekil 2.2 (b)'de taramalı elektron mikroskobu (TEM) (ing. SEM) sayesinde elde edilmiş görüntü bulunmaktadır. Şekil 2.2 (a) 'daki görüntü altın tasarım olarak kabul edildiğinde, Şekil 2.2 (b)'de bulunan görüntüde Truva atı eklendiği açıkça

gözükmektedir. Tersine mühendislikte ana zorluklardan en önemlileri, altın devreye ve şüphelilere ait görüntülerin en yüksek çözünürlük ve temizlikte elde edilmesidir.

Çalışma [48], donanımlar üzerinde yapılan tersine mühendislik çalışmalarını, hem yasal hem de yasadışı amaçlar için bir araç olarak ele almaktadır. Tersine mühendislik, entegre devrelerde IP ihlali ve kötü niyetli manipülasyonların tespitini desteklerken, aynı zamanda saldırganlara tasarımları çalmak veya Truva atlarını yerleştirmek için gerekli bilgileri sağlamaktadır. Bu çalışmanın en önemli katkısı, tersine mühendisliğin karmaşıklığını bilimsel olarak ölçmek için çeşitli açık araştırma soruları ortaya koyması ve hem teknik hem de insan faktörlerinin nicel değerlendirmesini yapmasıdır. Bu Tersine mühendisliğin teknik ve psikolojik perspektiflerden bütüncül olarak ele alınmasını önererek, gelecekteki araştırma yönleri için yeni bakış açıları sunmaktadır.



Şekil 2.2: (a):Tasarımdaki Görünüm , (b):Elektron mikroskobuyla görünüm.

Çalışma [49], truva atlarının tespiti için spesifik makine öğrenmesi yöntemlerine (SVM; K-means) odaklanırken, Çalışma [50] daha geniş bir bakış açısıyla görüntü işleme ve makine öğrenmesi tekniklerini entegre ederek genel donanım güvenliği sağlama amacını taşımaktadır. Bu makalede, altın devre olarak kullanılan ISCAS'89 devrelerinin GDS dosyalarından elde edilen bilgilerle her bir katman için ikili tabanda (ing. binary) resimler üretilmektedir. Bunun için resimdeki pikseller 0 ve 1 olarak işlenirler. Böylece referans

olarak kullanılacak devreye ait görüntüler elde edilmiş olur, ama test için yüzlerce devre görüntüsüne daha ihtiyaç vardır. Bunun için de elde edilen görüntüye rastgele gürültüler eklenir. Böylece varyasyon eklendiği ifade edilmektedir. Bu makaledeki problemlerden biri gerçek TEM resimleri kullanmamasıdır. Sentetik resimler kullanırken, varyasyon etkisini random gürültü ekleyerek yansıttığını söylemesi ve bunu detaylandırmaması da makalenin ikinci geliştirilmesi gereken kısmıdır. İşleri zorlaştırmak için, üç farklı ve tespiti oldukça zor olan Truva atı tipi eklenmiş ve her birinde sınıflandırma yöntemiyle başarıyla Truva atı tespit ettiği belirtilmiştir. Seçilen 3 farklı Truva atı tipi; transistörler arasındaki bağlantıyı silme, yeni bağlantı ekleme ve bağlantı boyutunu (ing. wire width) değiştirmedir.

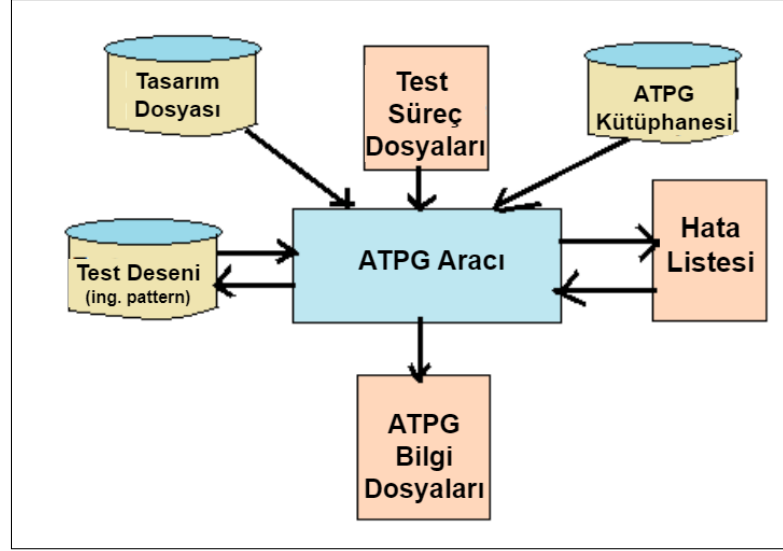
Çalışma [50], geniş kapsamlı bir güvenlik yaklaşımı sunmasının yanında, literatürde bulunan tersine mühendislik çalışmalarını ve görüntüleme cihaz/tekniklerini detaylı olarak incelemektedir. Farklı görüntüleme tekniklerinin (TEM, X-ray, DSLR kamera) çözünürlük kapasitelerini ve bu tekniklerin donanım güvenliği ve güvencesi bağlamında nasıl kullanıldığını anlatmaktadır. Makine öğrenmesi yöntemlerini de tüm aşamalarına göre ayrı ayrı incelemiş ve makale sonunda karşılaştırma tablosu ile özetlemiştir. Tersine mühendislik alanı için oldukça önemli bir araştırmadır.

2.1.2. Mantıksal Test Yöntemleri

Mantıksal testler (ing. logical tests), devreye belirli test vektörleri uygulayarak Truva atlarının tetiklenmesini ve ortaya çıkmasını amaçlar. Bu teknik, Truva atlarının nadiren karşılaşılan mantıksal koşullarda etkinleştiği varsayımıyla çalışmaktadır. Mantıksal test yöntemleri, devrelerin işlevselliğini test ederek anormallikleri tespit etmeye çalışırlar. Bu testler, devrenin güvenli ve güvenilir olduğunu doğrulamak için kritik öneme sahiptir. Bu testler, Truva atı tespitinin yanı sıra, devrenin genel güvenlik değerlendirmesinde de kullanılmaktadır [51].

Truva Atları genellikle, normal çalışma sırasında nadiren karşılaşılan veya belirli mantıksal koşullar altında etkinleşecek şekilde tasarlanırlar. Mantıksal testler, bu nadir koşulları bilinçli olarak oluşturarak olası Truva atını tetiklemeyi amaçlar [51–53].

Şekil 2.3 Otomatik Test Deseni Üretimi (OTDÜ) (ing. Automatic Test Pattern Generation) aracının çalışma sürecini ve bileşenlerini göstermektedir [51]. Tasarımı içeren



Şekil 2.3: ATPG Araçlarının Çalışma Süreci.

netlist dosyası, test prosedürü dosyasını ve OTDÜ kütüphanesini kullanarak test desenlerini oluşturur. Bu desenler, devrenin çeşitli çalışma koşullarını simüle etmektedir. Oluşturulan test desenleri, hata listesindeki hataları tespit etmek için kullanılır, hata simülasyonu gerçekleştirilir ve hangi hataların tespit edildiği belirlenir. OTDÜ süreci tamamlandığında, üretilen test desenleri ve tespit edilen hatalar hakkında bilgi dosyaları oluşturulur.

Otomatik test deseni üretim araçları kullanılarak, olası Truva atlarını etkinleştirme olasılığını maksimize eden özel test vektörleri oluşturulur [54]. Bu test vektörleri, devrenin düşük olasılıkla kontrol edilen ve gözlemlenebilen kısımlarına odaklanırlar. Bu kısımlar, saldırganlar tarafından Truva atının tetikleyici (ing. trigger) noktaları olarak seçilir ve test desenleri bu noktaların etkinleştirilmesi üzerine yoğunlaşır.

Test desenleri uygulandıktan sonra, devrenin çıktıları ve iç sinyalleri dikkatle izlenir ve analiz edilir. Beklenen davranış ile gözlemlenen davranış arasındaki herhangi bir sapmanın, Truva atının varlığını gösterdiği düşünülür. Bu sapmalar, devrede anormal aktiviteler veya beklenmedik sinyal değişiklikleri şeklinde kendini gösterebilmektedir.

Dikkate alınan nadir düğümlerin sayısına bağlı olarak aday tetikleyici kombinasyonlarının sayısı üstel bir bağımlılığa sahiptir ve bu sayı oldukça fazladır. Bu nedenle, listedeki her hatayı sıralı olarak ele almak mümkün değildir. Bu nedenle çalışma [54], test vektörleri üretimi için genetik algoritma (ing. genetic algorithm) ve tatmin edicilik testi (ing. SAT) tabanlı yöntemlerini birleştirmektedir. Mantık şu şekildedir: Uygu-

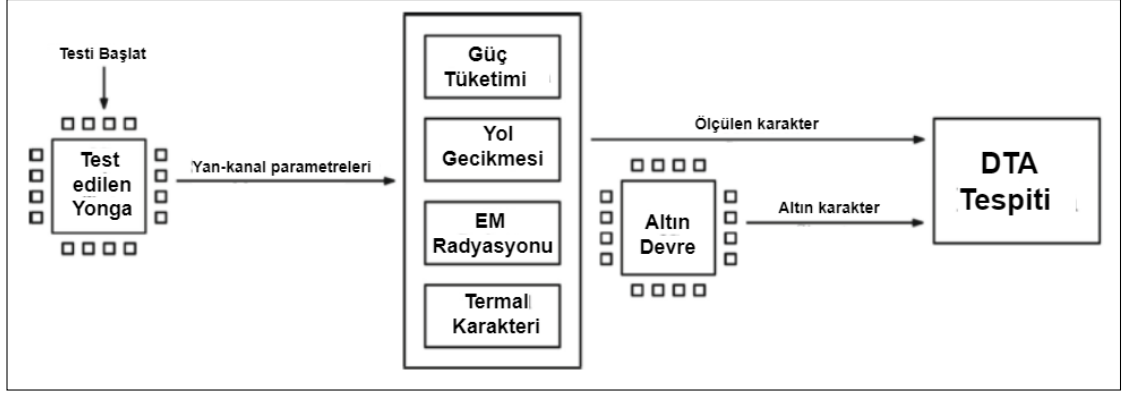
laması kolay tetikleyici koşulların çoğu ve uygulaması zor olan tetikleyici koşulların önemli bir kısmı, makul bir yürütme süresi içinde genetik algoritma tarafından tespit edilecektir. Kalan çözilemeyen tetikleyici vektörleri dikkate alınmalıdır [52]. Tatmin edicilik testi uygulanması için bahsedilen kalan tetikleyici koşullar girdi olarak işlenir. Ardından koşullardan herhangi biri tatmin edici bulunursa (ing. boolean satisfiability), koşulu sağlayan ilgili test vektörü oluşturulur. Aksi takdirde, ilgili koşul çözilemez olarak ilan edilecektir.

Avantajlar ve dezavantajlar açısından değerlendirdiğimizde, mantıksal test yöntemleri oldukça açık ve kolayca ayırt edilebilir olmasıyla öne çıkmaktadır. Ancak, bazı zorlukları da beraberinde gelmektedir. Mantıksal test yöntemleri, dışarıdan etkinleştirilen Truva atlarını tetikleyememekte ve yalnızca işlevsel Truva atlarını gözlemleyebilmektedir. Bu durum, yöntemin sınırlamaları arasında sayılabilir. Bir diğer dezavantaj ise kombinasyonel Truva atlarına nazaran, ardışık Truva atlarını tetikleyecek koşulları sağlamanın oldukça zor olmasıdır.

2.1.3. Yan Kanal Analizi Yöntemleri

Yan kanal analizi (ing. Side-channel analysis), devrelerin enerji tüketimi, elektromanyetik yayılım veya sıcaklık gibi yan kanal sinyallerini ölçerek Truva atı tespiti yapar. Bu yöntemler, Truva atlarının yan kanal sinyallerinde oluşturduğu değişiklikleri tespit etmeyi hedefler. Yan kanal analizi (YKA) yöntemleri, genellikle iki aşamadan oluşur: veri toplama ve veri analizi. Veri toplama aşamasında, devre çalışırken yan kanal sinyalleri toplanır. Bu sinyaller, devrenin normal çalışma modunda veya belirli test vektörleri uygulanarak toplanabilir. Veri analizi aşamasında ise, toplanan sinyaller analiz edilerek anormal davranışlar ve Truva atı varlığı tespit edilir (Şekil 2.4).

Bu tür tekniklerin Truva atları tespiti için tercih edilmesinin başlıca nedeni, devrenin çalışmasını kesintiye uğratmamaları ve devrenin işleyişini bozmamalarıdır. Modern yüksek frekanslı devreler için fiziksel parametrelerin doğru ölçümü son derece zor ve pahalı hale gelirken, transistörlerin boyutlarının sürekli küçülmesi nedeniyle artan süreç varyasyonları ölçümlerde yüksek oranda gürültüye neden olmaktadır. İki aynı devrenin yan kanal imzaları, aynı giriş vektörleri uygulandığında bile farklı olabilmektedir. Bu farklılığın Truva atı etkisi yüzünden olması gerekirken, üretim varyasyonları nedeniyle



Şekil 2.4: YKA Yöntemlerinin Çalışma Süreci.

olmaktadır. Üretim varyasyonları kolayca Truva atı etkisini maskeleymektedir. YKA yöntemlerinin en büyük problemi, süreç varyasyonunu izole etmektir. Transistör boyutları küçüldükçe varyasyonun ölçümlere olan etkisi arttığından, izole etmek gittikçe zorlaşmaktadır.

Şekil 2.4’de YKA yöntemlerinin çalışma süreci gösterilmiştir. Veri toplama kısmında; test edilen yonga üzerinde ilgili girdiler verilerek test başlatılır, sonrasında gözlenmek istenen YKA yöntemine dair ölçümler yapılır. Veri analizi kısmında ise toplanan ölçüm değerleri ile altın devreye ait ölçüm sonuçları kıyaslanarak Truva atı tespiti gerçekleştirilir.

Avantajlar ve dezavantajlar açısından değerlendirildiğinde, YKA yöntemin bazı önemli artıları ve eksileri bulunmaktadır. Avantajlarından biri, devrelerde gözle görülür bir işlev bozukluğuna neden olmayan Truva atları için etkili olmasıdır. Bu, özellikle sinsi bir şekilde yerleştirilmiş Truva atlarının tespitinde büyük bir avantaj sağlamaktadır[2]. Ancak, yöntemin bazı dezavantajları da bulunmaktadır. Modern üretim teknolojilerindeki büyük süreç varyasyonları, Truva atlarının tespit edilmesini zorlaştırmakta ve yöntemin etkinliğini azaltmaktadır.

Tablo 2.1’de YKA yöntemlerinden dört tanesi seçilmiş ve karşılaştırmalı olarak incelenmiştir. İlk sütunda bulunan başlıklardan; YKA parametresi kullanılan yöntemi, süreç varyasyonu başlığı kullanılan varyasyon modelinin gerçeğe etkisini, ölçüm gürültüsü başlığı ise yan kanal ölçümleri alınırken çevresel faktörlerin etkisini göstermektedir. Deney ortamı olarak iki yöntemde FPGA birinde ASIC kullanılmış, diğer çalışma ise simülasyon ortamında doğrulanmıştır. Çalışmalardan ikisinde, yaygın olarak kullanılan

devrelerden AES kriptotarif devresi, birinde ISCAS85 devrelerinin en küçüğü olan c432 devresi, bir diğesinde ise DLX işlemcisi kullanılmıştır. Kullanılan Truva atı çeşitleri ise tablonun son satırında yer almaktadır.

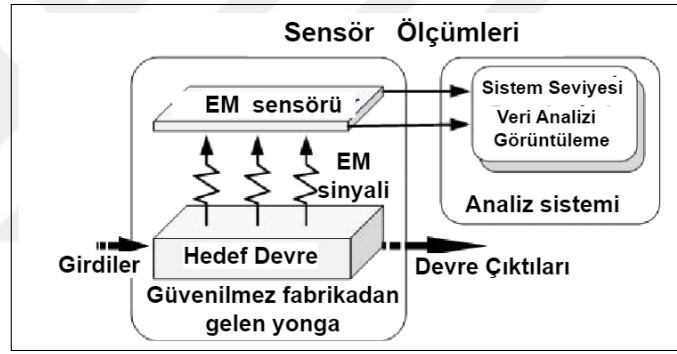
Tablo 2.1: YKA yöntemlerinin Karşılaştırılması.

	[34]	[55]	[21]	[18]
YKA parametresi	Akım	EM	Güç	Gecikme
Süreç Varyasyonu	Düşük	Yüksek	Düşük	Yüksek
Ölçüm Gürültüsü	Düşük	Yüksek	Yüksek	Düşük
Deney Ortamı	FPGA	FPGA	ASIC	Simulasyon
Devreler	DLX işlem- cisi	AES (Trusthub)	AES (350nm)	c432 (IS- CAS'85)
DTA çeşitleri	7 farklı tipte 96 tane	11 tane (Trusthub)	2 tane	2 tane

YKA yöntemlerinde literatürde en yaygın kullanılanlardan biri güç ve akım tüketimine göre [56, 57], bir diğere ise yol gecikmesine göre Truva atı tespittir. Güç tüketimine göre Truva atı tespitinde, devre bölümlendirilerek her bir bölümün güç tüketimi karakterize edilir. Bunun içinde genellikle Halka Osilatörü (HO) yapısı kullanılmaktadır. HO kullanarak Truva atlarının tespiti, özellikle entegre devrelerdeki frekans değışikliklerini izleyerek Truva atlarını belirlemek için kullanılan etkili bir yöntemdir. Bu yöntemde, HO'nun frekansı, devredeki yük veya bozulmalara bağılı olarak değışmektedir. Truva atları, devreye ek yük getirerek veya belirli yolları değıştirerek HO'nun frekansında gözlemlenebilir değışikliklere neden olmaktadır. [20, 37, 58] çalışmalarında, bu yöntemin farklı yaklaşımlarını ve uygulamalarını detaylandırarak donanım güvenliği alanında önemli katkılar sağlamıştır. Bu tür yöntemlerin daha güvenilir ve verimli hale getirilmesi için diğere analiz teknikleri ile birlikte kullanılması önerilmektedir [33, 59].

2.1.4. Çalışma Sırasında Tespit Eden Yöntemler

Çalışma sırasında (ing. run-time monitoring) tespit eden yöntemler, devrelerin normal çalışma sırasında Truva atlarını tespit etmeyi amaçlar. Bu yöntemler, genellikle dinamik analiz ve sürekli izleme gerektirir [60]. Çalışma sırasında tespit eden yöntemler, genellikle donanım performansını izlemek için ek modüller gerektirir. Bu modüller, devrelerin normal çalışması sırasında sürekli veri toplar ve bu verileri analiz ederek Truva atının varlığını tespit eder [61, 62]. Örneğin, devre üzerindeki sıcaklık değişiklikleri, güç tüketimi ve performans ölçümleri izlenerek anormal durumlar belirlenebilir. Şekil 2.5’de gösterilen çalışma [60], sol tarafta veri toplama işlemini yapmaktadır, devrenin çalışması sırasında oluşan elektro manyetik değerler sensörler aracılığıyla toplanır. Şeklin sağ tarafında yer alan analiz sistemleri ise veri analiz kısmını içermektedir.



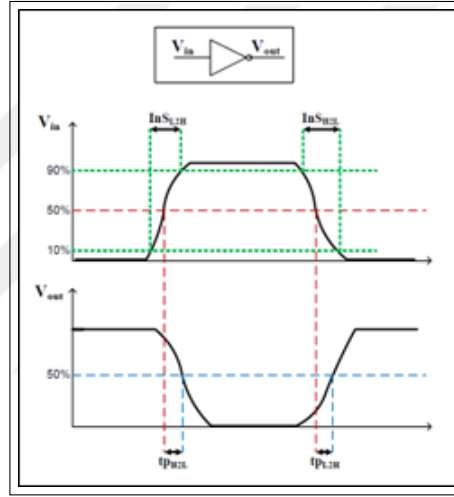
Şekil 2.5: Çalışma sırasında DTA tespitinin gösterimi.

Literatürde yaygın olarak referans alınan [63] çalışmasında, Truva atlarının çalışma sırasında tespiti için devreye olabildiğince düşük ek yük getiren üç yenilikçi yöntem önerilmektedir. Bu yöntemler, modern sistemlerde zaten mevcut olan termal sensörleri kullanarak, Truva atlarının etkinleştirilmesiyle oluşan güç/termal profillerdeki sapmaları tespit etmeyi amaçlar. Önerilen üç yaklaşımdan ilki, termal sensörlerden alınan bilgiler ve hipotez testi kullanılarak karar verir, ve oldukça basit tasarlanmıştır. İkincisi, sensörler arasındaki korelasyonu kullanarak devrenin termal profilini takip eder. Son ve en gelişmiş yöntem, entegre devrelerde kaçak güç (ing. leakage power) tüketiminin izler. Entegre devrelerde, transistörlerin kapalı olduğu durumlarda bile oluşan küçük akımlardır. Bu nedenle, kaçak güç termal profilde değişikliklere neden olabilmektedir. Sistemin [63] dinamik modeli, entegre devrenin zaman içindeki davranışını tanımlayan

matematiksel bir modeldir. Kaçak güç, bu modele dahil edilerek, termal profildeki değişikliklerin daha doğru bir şekilde izlenmesi sağlanır. Böylece gerçek zamanlı olarak sıcaklık değişiklikleri takip edilmektedir.

2.2. İlgili Çalışmalar: Yol Gecikmesi Kullanan Yöntemler

Yayılma gecikmesi (ing. propagation delay), devrelerde sinyallerin bir noktadan diğerine iletilme süresini ifade eder. Bu, bir kapının giriş ve çıkış sinyalleri arasındaki gecikmeyi ölçmek için önemli bir parametredir. Şekil 2.6 [?], bir DEĞİL (ing. NOT) kapısının giriş ve çıkış voltaj dalga formlarını ve yayılma gecikmesini göstermektedir.



Şekil 2.6: DEĞİL Kapısına ait Yayılma Gecikmesinin Ölçümü.

Kapı bir DEĞİL kapısı olduğundan, giriş sinyali düşük voltajdan yüksek voltaja geçiş yaptığında, çıkış sinyali yüksek voltajdan düşük voltaja geçiş yapar. Giriş ve çıkış sinyallerinin yüksek voltajın yarısına ulaştığı anlar arasındaki zaman farkına kapının yayılma gecikmesi (kapı gecikmesi) denir. Çıkış sinyalinin geçişine bağlı olarak, yayılma gecikmesi düşükten yükseğe yayılma gecikmesi $L2H$ veya yüksektan düşüğe yayılma gecikmesi $H2L$ olarak adlandırılır.

Yol gecikmesi kullanan Truva atı tespit yöntemleri, devrelerin yol gecikmelerini ölçerek ve analiz ederek, gecikme değerlerinde oluşan anormallikleri yani Truva atlarını tespit etmeyi hedefler. Bu bölümde, bu yöntemlerin farklı yaklaşımları, ve birbirleriyle karşılaştırmaları incelenecektir.

Tablo 2.2: Yol Gecikmesi Yöntemlerinin Karşılaştırması.

Çalışmalar	Devreler	DTA Modeli	Varyasyon Parametreleri	Varyasyon Modeli
[16] (2018)	ISCAS'89 & DES	yük = 1xor	yol-gecikmesine eklenmiş	yonga-içi %20 yongalar arası %8
[19] (2021)	Trusthub	Trusthub	rastgele %5	-
[65] (2021)	ISCAS'85 & 89	yük = 1xor	TSMC-90 nm	-
[34] (2022)	DLX işlemci	3 farklı 32bit model	FPGA (55nm)	gerçek cihaz üzerinde
[46] (2023)	ISCAS'85	yük=1xor	ITRS2011 - 45nm	yonga-içi %50 yongalar arası %50

Yol gecikmesi ölçüm yöntemleri, yol gecikmesi ve istatistiksel zamanlama analizi (ing. STA) gibi teknikleri içermektedir. Yol gecikmesi ölçüm yöntemleri ise, belirli bir yol üzerindeki toplam gecikme değerini ölçerek çalışır [17]. Gecikme ölçümleri için İstatistiksel zamanlama analizi ise, devrelerin gecikme değerlerini hassas modellemeler kullanarak analiz eder [44, 64].

Tablo 2.2'de beş farklı yol gecikmesi yöntemi incelenmiştir. Kullandıkları devre modelleri, Truva atı modeli ilk iki sütundadır. Son iki sütunda bulunan varyasyon parametreleri ve modelinde bahsedilenler; [34] çalışması 55nm FPGA üzerinde yapıldığından varyasyon etkisi gerçektir, [65] çalışmasında TSMC-90nm ait veriler, [46] çalışmasında ise ITRS 2011 yılındaki raporundan gelen 45nm [66] ait veriler yongalar arası varyasyon etkisi %50 olacak şekilde kullanılmıştır. İlk iki satırda bulunan çalışmalar olan [16] ve [19], varyasyon modeli olarak etkisi daha az olan ve zayıf sayılabilen yöntemleri kullanmışlardır.

Gecikme tabanlı birçok Truva atı tespit yöntemi literatürde bulunmaktadır. Çalışma [67]'de gecikme tabanlı testlerde, test edilen devreye farklı giriş desenleri uygulanmaktadır. Elde edilen gecikmeler, gecikme parmak izi adı verilen çok boyutlu bir vektör olarak toplanmakta ve ardından donanım Truva atı tespiti için kullanılmaktadır. Bu yöntem sadece %7,5 değişim varsaymakta ve yük kısmı tek bir XOR kapısı olan ve tespiti zor olan sayaç tabanlı bir Truva atı'nı %36 başarı oranıyla tespit edebilmektedir.

Çalışma [18]'de varyasyondan kaynaklanan değişimlerin etkisini sönmölemek için, benzer topolojiye sahip yollar olan simetrik yollar çıkarılmaktadır. Bu yöntem, bu yol gecikmelerinde simetri bozulmasını bir Truva atı işareti olarak kullanmaktadır ve altın model kullanıma ihtiyaç duymamaktadır. Ancak yöntemin dezavantajı, çok küçük devrelerde test edilmesi ve ölçeklenebilirlik sorunu olmasıdır. Bir diğer sorun, yonga-içi değişimlerin topolojik olarak simetrik yolları farklı şekilde etkileyip simetriyi bozabilmesidir.

Başka bir gecikme tabanlı yaklaşım [17], devre kenarına yerleştirilen Truva atının yol gecikmesine olan etkisini artırmak için en kısa yolları tercih etmektedir. Ancak, tüm yongaların aynı partiden, yani tamamen aynı tasarıma sahip şekilde, gelmesi gerektiğini savunmaktadır. Ve üretimden gelen yongaların sadece bir kısmının seçilerek Truva atı yerleştirilebileceğini göz ardı etmektedir. Bu durumda, tek bir yonganın kapsamlı analizi yeterli olmaktadır. Analiz sonucunda Truva atı tespit edilirse, ilgili partiden gelen tüm yongaların Truva atı içerdiği sonucuna ulaşılır.

Çalışma [16]'de bir yol gecikme tabanlı tespit sunmaktadır. Çalışmada tüm devreyi kapsayan bir yol kümesi çıkarılmakta ve yol gecikmeleri, geleneksel SLE kapı gecikme modeli [68] kullanılarak hesaplanmaktadır. Bu model, gecikmeye doğrudan uygulanan %8 yonga içi ve %20 yonga arası değişimler varsaymaktadır. Tüm olası yol çifti kombinasyonlarını numaralandırarak sıralamakta, ve ardından, test devresinde yol gecikme sırasındaki değişikliklerin sayısını karşılaştırarak Truva atı yerleştirmesini tespit etmektedir. Eğer test devresinde altın devrelerden çok daha fazla değişiklik varsa, Truva atı varlığı sonucuna varmaktadır. Ancak yöntemin dezavantajları, altın devrelere ihtiyacı olması dışında, süreç varyasyonunun doğrudan yol gecikmesi üzerine uygulanması ve performansın sadece rastgele seçilen üç devre kenarında doğrulamasıdır.

Bu çalışmada [69], yerleştirilen Truva atı tarafından işgal edilen parasitik kapasitans

ölçülmektedir. Bazı Truva Atlarının yol gecikmesini etkilemediğini ancak ekstra kapasitans nedeniyle tespit edilebileceğini tartışmaktadır. Altın model ve Truva atı etkinleştirme gereksinimleri eleştirilmektedir ve bu nedenle, pasif Truva Atlarının altın çip gerektirmeyen tespiti önerilmektedir. Yöntem, ISCAS'85 devrelerinde doğrulanmaktadır. Ancak, tasarımı değiştirmeleri ve kapasitans algılama devresi eklemeleri gerekmektedir. Ayrıca kullanılan varyasyon modelinin detayları, sonuçları önemli ölçüde etkilediğinden daha kapsamlı olarak açıklanmaya ihtiyaç duymaktadır.

Bu çalışmaya benzer şekilde, [19]'deki yazarlar da altın model gerektirmeyen gecikme tabanlı yan kanal analizi kullanarak Truva atı varlığını tespit etmeye çalışmaktadır. Benzer şekilde, bir XOR yük ve 4 girişli ve VE (ing. AND) kapısı tetikleyici parçası olan minimal bir Truva atı eklendiğini varsaymaktadır. Yöntem, statik zamanlama analizinden yararlanan makine öğrenimine dayanmaktadır. Üretim sonrası test aşaması SPICE kullanılarak simüle edilmektedir. Elde edilen tespit oranı ortalama %98 ile umut vericidir. Ancak, test aşaması için modellenen süreç varyasyonu miktarı %5 olup, modern süreç varyasyonlarına kıyasla daha az miktardadır. Varyasyon miktarının azalması, Truva atı tespit oranını artırmaktadır. Ayrıca, tüm üretilen yongalara Truva atı yerleştirildiği varsayılmaktadır. Ancak böyle bir durumda tek bir çipin kapsamlı veya tersine mühendislik yöntemiyle test edilmesi bile Truva atı varlığını kolayca ortaya çıkaracaktır. Genellikle Truva atının kolayca tespitini engellemek için üretim sırasında devrelerin sadece bir kısmına Truva atı eklenmesi tercih edilmektedir [33].

2.2.1. Kullanılan Devreler

Test devreleri, Truva atlarıyla ilgili araştırmalar için hayati öneme sahiptir çünkü farklı türde ve ölçekteki saldırıları simüle etme imkanı sunmaktadırlar. Bu kıyaslamalar sayesinde, tespit ve önleme yöntemlerinin gerçek dünya senaryolarında nasıl performans gösterdiği değerlendirilebilmektedir. Aşağıdaki devrelerin tamamı açık kaynaklıdır, gerçek dünya uygulamalarında Truva atlarının olası etkilerini göstermek için kullanılmaktadırlar. Truva atları, farklı modüller arasındaki veri akışını manipüle ederek veya işlemci komutlarını değiştirerek entegre edilmektedir. Truva atlarının etkilerini ölçmek ve karşılaştırmak için kullanılan test devrelerinin başlıcaları şunlardır:

- ISCAS'85 [38] ve ISCAS'89 [70] Test Devreleri: Bu devreler, başlıca toplayıcı/

çarpıcı devreleri, dijital sinyal işleme ve iletişim sistemleri gibi uygulamaları simüle eder. Truva atı tespit tekniklerinin doğruluğunu değerlendirmede kullanılan başlıca devrelerdendir. Truva atları, bu devrelerin çeşitli bölümlerine entegre edilerek, tespit algoritmalarının etkinliği test edilmektedir. Devreler verillog dosyası olarak paylaşılmıştır.

- Trust-HUB [22] Test Devreleri: Güvenilir donanım tasarımı üzerine odaklanmış olan bir platformdur. Truva atları sınıflandırma tablosuna göre etiketlenmiş çeşitli senaryolar sunmaktadır. Truva atı tespit tekniklerinin doğruluğunu değerlendirmede kullanılan en yaygın devrelerdir. Bu devreler, akademik araştırmacılara ve endüstri uzmanlarına, DTA tiplerinin tespiti ve önlenmesi konusunda geniş bir test yelpazesi sağlamaktadır. Devreler genellikle verillog dosyası olarak paylaşılmıştır. Üretim aşamasında eklenen Truva atlarının testi için .def dosyaları bulunmaktadır.
- Açık Kaynaklı [71] Test Devreleri: Makine öğrenmesi teknikleriyle kullanılmak için geliştirilmişlerdir. Makine öğrenmesi yöntemleri oldukça fazla sayıda devreye ihtiyaç duyduğundan, çeşitli alanlarda (ASIC, FPGA) sentetik devreler üretilmiştir. Devreler verillog dosyası olarak paylaşılmıştır.

2.2.2. Karşılaşılan Zorluklar

Literatürdeki en etkili yöntemlerden biri, yan kanal analizinin bir alt başlığı olan gecikme tabanlı Truva atı tespittir. Ayrıca, yıkıcı tespit yöntemlerinin aksine yongayı tahrir etmeden yaygın olarak kullanılan gecikme testleri ile uygulanabilmektedir. Gecikmeye dayalı Truva atı tespitinin başarısı Truva atının boyutuna bağlıdır çünkü Truva atı ne kadar büyükse o kadar fazla gecikme değerine sahiptir. Başlıca dezavantajı ise Truva atı devresinin gecikmeye etkisini rahatlıkla gözardı edebilen süreç varyasyonlarıdır.

Gecikme tabanlı yöntemlerin karşılaştığı zorlukları daha ayrıntılı ve anlaşılır şekilde maddeleyelim:

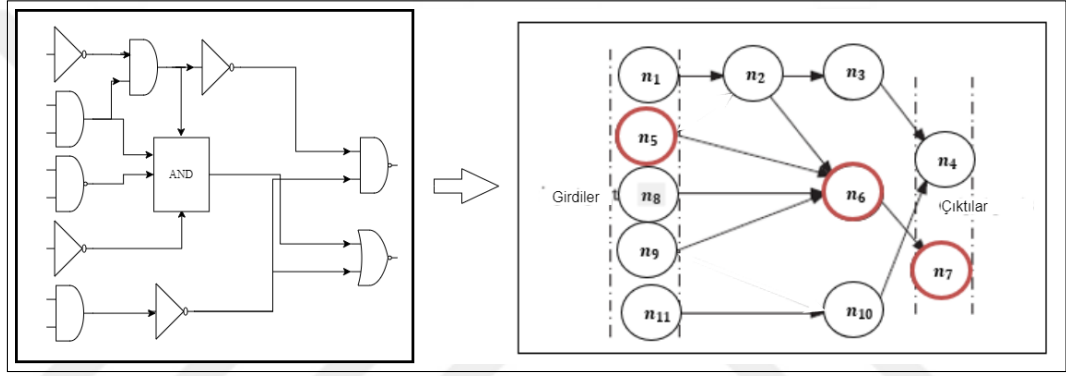
- Artan Süreç Varyasyonu ile Hassasiyetin Azalması: YKA tabanlı yöntemlerin hassasiyeti, üretim süreçlerindeki varyasyonlar nedeniyle azalabilir. Yani, çiplerin üretimindeki küçük farklılıklar, YKA'nın güvenilirliğini etkileyebilir ve yanlışma olasılığını artırabilir.

- Küçük DTA'ların Tespit Edilememesi: Üretim süreçlerindeki varyasyon etkilerinin YKA değerlerinden (ör: yol-gecikme değeri) daha fazla olması durumunda meydana gelebilir. Bu, bazı küçük boyutlu Truva atlarının yanlışlıkla zararsız olarak değerlendirilmesine yol açabilir.
- Altın Model/Devre Gereksinimi: YKA teknikleri, karşılaştırma yapmak için bir "altın model/devre"ye ihtiyaç duymaktadır. Altın model, analiz edilecek sistemin tamamen güvenli ve doğru çalışan bir referans versiyonudur. Bu model olmadan, tespit edilen anormalliklerin doğruluğunu değerlendirmek oldukça zordur.
- DTA Aktifleştirme: Rastgele Truva atı örneklerini etkinleştirmek ve etkilerini gözlemlemek son derece zor olabilir. Truva atlarının etkinleştirilmesi, genellikle belirli koşullar altında gerçekleşir ve bu koşulları simüle etmek zor olabilir. Ancak, YKA teknikleri, Truva atlarının etkinleştirilmesine gerek kalmadan analiz yapmaktadır, adını da buradan almaktadır.
- Ek Devre Gereksinimi Olmaması: Harici ölçüm kullanan YKA teknikleri, analiz edilecek donanıma ek devreler eklemeyi gerektirmezler. Bu, donanımın orijinal yapısını değiştirmeden analiz yapabilmeyi sağlar, ancak bazen ek devreler eklemek analiz hassasiyetini artırabilir.
- Mevcut Yaklaşımların Zaman Alıcı Olması: Literatürdeki mevcut YKA yöntemlerinin çoğu, zaman alıcı ve belirli bir test devresi ya da truva atı grubu için tasarlanmışlardır. Bu yöntemler, detaylı veri toplama ve analiz süreçleri gerektirdiğinden, büyük ölçekte uygulanmaları maliyetli olabilir.

3. Yönteme Hazırlık

3.1. Devre Çizgesi

Devreyi ifade etmek için yönlü çevrimsiz çizge (ing. direct acyclic graph) yapısı kullanılmaktadır, bunu devre çizgesi olarak adlandırmaktayız. Devredeki her bir mantıksal kapı, çizge üzerinde düğümle ifade edilmektedir. Devredeki iki mantıksal kapıyı birbirine bağlayan kablo, çizgede kenar adıyla ifade edilmektedir. Şekil 3.1 de sol tarafta devre şekli, sağ tarafta devre çizgesi gösterilmektedir.



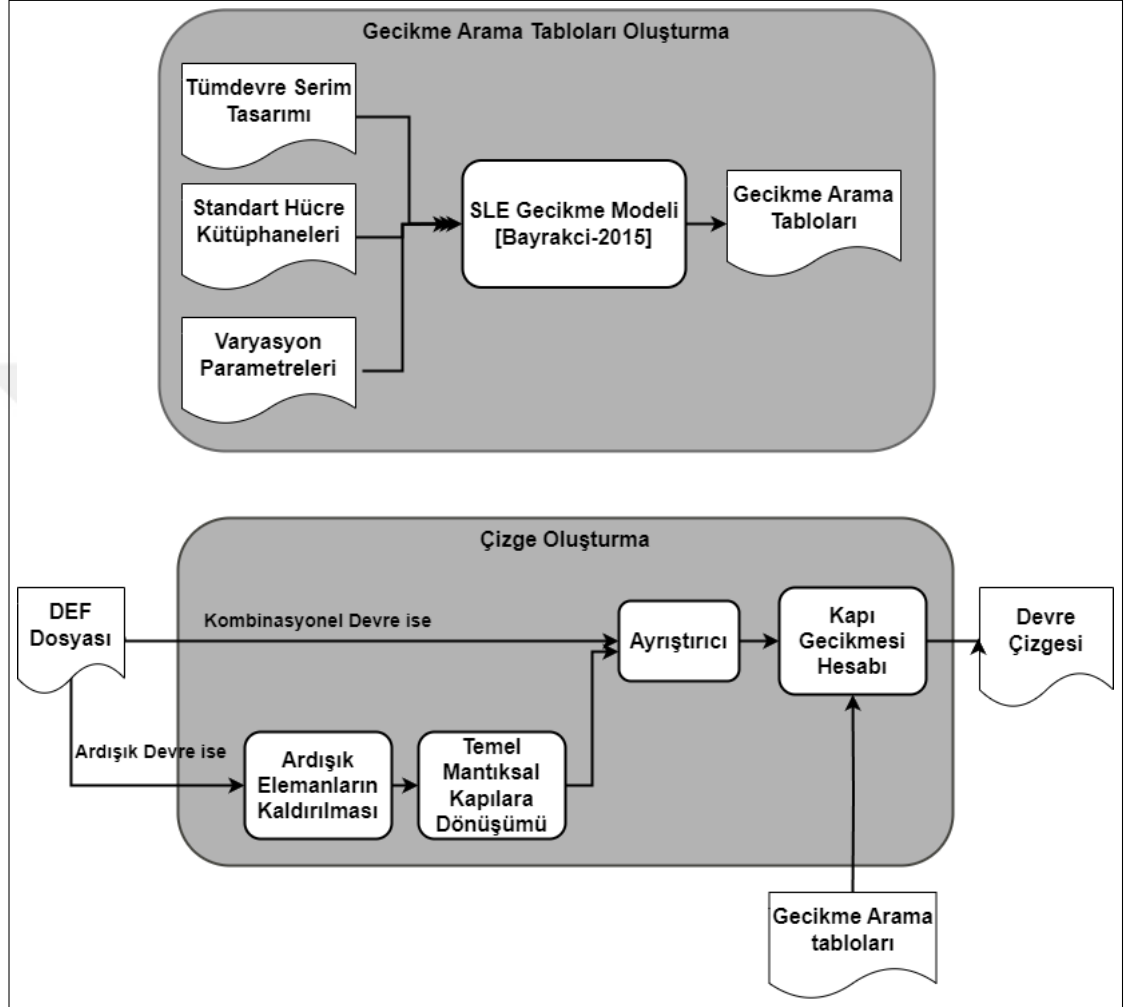
Şekil 3.1: Devre Çizgesi Oluşturma.

Devredeki herhangi bir yol birincil girişten (ing. Primary Input) başlamakta, kapılardan (düğümlerden) geçmekte ve birincil çıkışta (ing. Primary Output) sona ermektedir. Şekil 3.1 de kırmızı renkli düğümler, örnek yola ait düğümleri ifade etmektedir. Örnek yol; $\langle n_5, n_6, n_7 \rangle$ şeklinde ifade edilmektedir. n_p ve n_q düğümleri arasındaki kenar $e_{p,q}$ olarak gösterilir.

Devre çizgesinde her bir düğüm çeşitli özelliklere sahip olmaktadır. Çizge üzerinde yapılan işlemlerde bu özelliklerden faydalanılmaktadır. Düğümlerde tutulan özellikler;

- mantıksal kapı türü (VE, VEYA, NOT kapısı gibi),
- komşu düğümler, ata düğümler,
- düğümün girdi/çıkı noktası olup olmadığı,
- yerleşim planından lokasyon bilgileri (Bölüm 3.2 de anlatılmaktadır.)

- kapı gecikmesi değeri (Bölüm 3.3 de anlatılmaktadır.)
- çıkış yükleme kapasitesi(ing. fanout), çıkış gerilimi değişim hızı (ing. output slope)



Şekil 3.2: Devre Çizgesi Oluşturma Aşamaları.

Devre çizgesinin elde edilmesi için gereken veri, sentezleme işlemi sonunda oluşturulan ve tümdevre serimi (ing. layout) bilgilerini içeren DEF uzantılı dosyadır. Kombinasyonel devreler için, ISCAS'85 devrelerine ait olan ve Texas A&M Üniversitesi'ndeki VLSI CAD grubu [22] tarafından sentezlenmiş olan DEF dosyaları kullanılmıştır. Ardışık devreler için, Trusthub veri kümesi içinde üretim başlığı [38] altındaki DEF dosyaları kullanılmıştır. Trusthub devreleri içinde başka bir başlık altında maalesef devrenin yerleşim planına ait dosyalar bulunmamaktadır.

Bu çalışmada tüm gecikme ölçümleri için Stokastik Lojik Efor (SLE) adı verilen hız-

lı bir kapı gecikme modeli kullanılmıştır [44]. Hassas transistör düzeyinde SPICE simülasyonları ile oluşturulan bu gecikme modeli, değişen devre parametrelerine göre gecikme varyasyonunu da yakalayabilmektedir. SLE gecikme modeli [44] standart hücre kütüphanesine ait bilgiler, tümdevre tasarımı ve ilgili üretim modeline göre tanımlanmış varyasyon parametrelerini girdi olarak kullanır. Devre çizgesi oluşturma, yol gecikmesi hesaplama gibi gecikme içeren tüm işlemlerde kullanılacak olan gecikme arama tablolarını oluşturur.

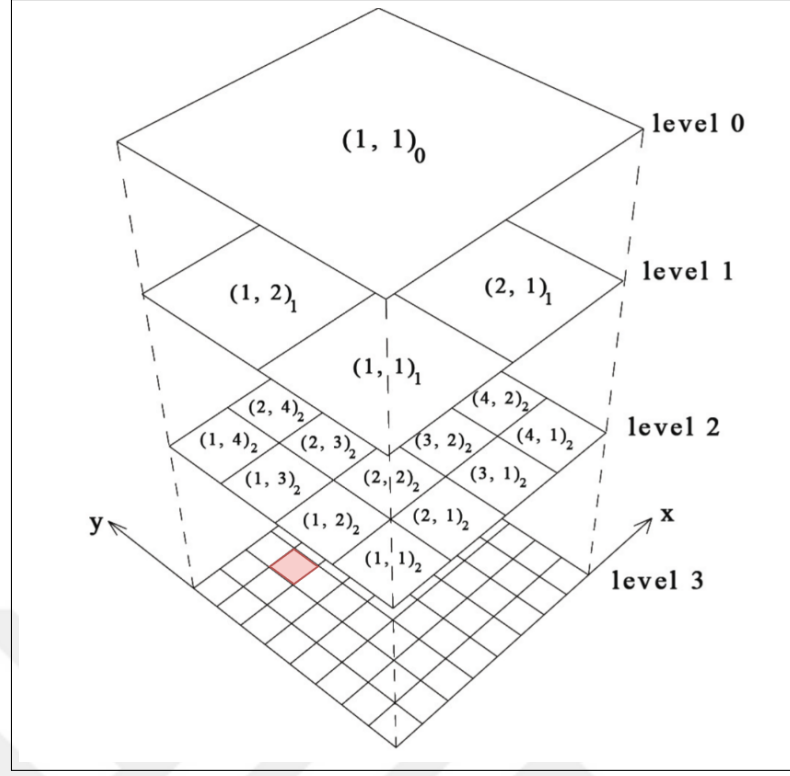
Üretim sonrası test aşaması olan Bölüm 5’den önce bir devre çizgesini oluşturmamız ve düğüm gecikmelerini atamamız gerekir, bu Şekil 3.2’deki Devre Çizgesi Oluşturma Bölümündedir. Devre tasarımı ayrıştırıcı ile çizge haline dönüştürülür, gecikme arama tabloları kullanılarak düğüm gecikmeleri hesaplanır ve çizgeye atanır.

3.2. Gerçekçi Varyasyon Modelimiz

Bölüm 2.2 kısmında ele aldığımız çalışmalardan olan [16], yol gecikmelerini hesaplamak için hızlı mantıksal efor (ing. logical effort) kapısı gecikme modelini [68] kullanmaktadır. Bu model Cui için uygundur çünkü varyasyonları rastgele parametreler yerine doğrudan gecikmeye uygulamaktadırlar. Oysa, biz devrenin gerçeğe daha yakın davranışını elde etmek için, rastgele süreç parametrelerine varyasyonlar uygulamaktayız. Bu nedenle, [16]’deki basit varyasyon modelini kullanamayız. Bölüm 3.2’deki varyasyon modelini kullanmaya ihtiyaç duymaktayız.

Bölüm 1.4’de varyasyonun yapısı ve genel özellikleri incelenmektedir. Konumsal korelasyon yararlanmak istediğimiz için, varyasyon modelimizde yonga-içi ve yongalar arası varyasyonu dikkate alan bir model kullanmamız gerekmektedir. Bu nedenle, [72]’te önerilen dörtlü ağaç modelini kullanmaktayız. Bu model konumsal korelasyonlarla birlikte hem yongalar arası hem de yonga-içi varyasyonları dikkate almaktadır. Fiziksel devre yerleşimi için farklı seviyeler kullanır ve yerleşimi her bir alt seviyede daha fazla sayıda ızgaraya böler. Varyasyon modelimizi daha gerçekçi ve hassas hale getirmek için, test devrelerimizin boyutuna ve [72]’te tasarlanan ızgara boyutuna göre, Şekil 3.3’de gösterilen dört seviyeli bir dörtlü ağaç modeli kullanmaktayız.

Her seviyedeki her bir ızgaraya tercih edilen varyasyon modeline göre rastgele bir değer atanır. Bu değer rastgele parametre değeri olarak adlandırılır. Bir transistör için



Şekil 3.3: Dördün Ağaç Yapısı.

hesaplanan rastgele parametre değeri, transistörün bulunduğu seviyedeki her bir ızgaraya karşılık gelen rastgele değerin toplanmasıyla elde edilir. Örneğin, Şekil 3.3’de kırmızı renkli gösterilen ızgarada bulunan transistörlerin rastgele parametre değerini hesaplamak için; 3.seviyede ızgara (2,3)’ün değeri, 2.seviyede (2,4)’ün değeri, 1.seviyede (1,2)’nin değeri ve 0.seviyede (1,1)’in değeri toplanır.

Böylece birbirine yakın transistörlerin rastgele parametreleri, birbirinden uzak transistörlere göre daha ilişkili değerlere sahip olurlar. Varyasyon modeliyle gerçekleştirmek istediğimiz konumsal korelasyonların etkisini yansıtmaktı, bu dördün ağaç yapısı sayesinde bunu elde etmiş olduk. Sonuç itibarıyla, her bir rastgele parametre için, her biri Şekil 3.3’de bulunan ayrı bir ızgaraya karşılık gelen rastgele değerler varyasyon modelimiz tarafından kullanılmaktadır. Bu rastgele değerler, 3.seviyeden 64 adet, 2.seviyeden 16 adet, 1.seviyeden 4 adet ve 0.seviyeden 1 adet olmak üzere toplam 85 adettir. Kullanılacak olan her bir parametre için 85 adet rastgele değer oluşturulur.

Gerçekçi bir varyasyon modeli elde etmek için, bu dördün ağaç modeli ile entegre devreler için [40]’de bildirilen gerçek 3σ varyasyonlarını ve [73]’de önerilen yongalar arası ve yonga içi varyasyon bileşenlerini eşit miktarda kullanarak birleştiriyoruz. Her

bir rastgele parametre için hesaplanan toplam varyasyon, yongalar arası ve yonga içi bileşenlerin toplamıdır (Denklem 3.1). Yonga-içi ve yongalar arası varyasyonlar %50 oranında kullanılmaktadır [33].

$$\sigma^2 = \sigma_{yongalar-arasi}^2 + \sigma_{yonga-ici}^2 \quad (\sigma_{yongalar-arasi}^2 = \sigma_{yonga-ici}^2) \quad (3.1)$$

[74] çalışmasında önerildiği gibi, devre gecikmesi üzerinde en etkili iki rastgele parametre transistör kapı uzunluğu (L_{eff}) ve eşik gerilimidir (V_t). Modelde, [66] 45nm teknolojisini kullandık ve [40]'de standart boyutlu mantık devreleri için bildirilen transistör geçit uzunluğu ve eşik gerilimi için sırasıyla %12 ve %20'lik bir $3\sigma/\mu$ varyasyon değerlerini kullanmaktayız.

Bu varyasyon modeli Monte Carlo tarzında örnekler üretmek için kullanılmaktadır. Her bir örnek üretilen bir yongayı temsil etmektedir. Her bir parametre için 85 rastgele değişken olmasından bahsetmiştik (her biri Şekil 3.3'deki bir ızgaraya karşılık gelir). İki rastgele parametre (L_{eff} ve V_t) varsaydığımızdan, her bir örnek (ing. sample), ortak olasılık dağılımından (ing. probability density function) çekilen 170 rastgele değişkenden oluşmaktadır. Bu 170 rastgele değişken, ilgili örnek için her bir transistörün L_{eff} ve V_t değerlerini hesaplamak için kullanılır.

3.3. Gecikme Ölçümü için Stokastik Lojik Efor Modeli

Stokastik Lojik Efor (SLE) [44] iyi bilinen lojik efor gecikme modelinin genişletilmesinden oluşmuştur. Kullandığımız SLE modeli, birim çevirici gecikmesini, parazitik ve mantıksal çaba gecikme bileşenlerini Şekil 3.2'de gösterildiği gibi devre içindeki rastgele parametrelerin bir fonksiyonu olarak varsaymaktadır. Burada $d_{n_r}^{SLE}(X)$ SLE yöntemi ile hesaplanan n_r mantık kapısının (düğüm) gecikmesini ifade etmektedir.

Formülasyonundaki τ birim çevirici gecikmesi (ing. unit inverter delay), p parazitik bileşen (ing. parasitic component), g mantıksal çaba (ing. logical effort) ve h elektriksel fan-out'dur. X devredeki tüm rastgele süreç parametrelerini ifade etmektedir. Örneğin, kapı uzunluğu ve eşik voltajı devredeki rastgele parametreler olarak alınırsa, $f(X)$ ortak olasılık yoğunluk fonksiyonundan (ing. pdf) çekilen bir örnek noktası X_i , devredeki her transistörün kapı uzunluğu ve eşik voltajı atamasını içermektedir.

$$d_{n_r}^{SLE}(X) = \tau(X)(p(X) + g(X)h) \quad (3.2)$$

SLE modeli istenen $\tau(X)$, $p(X)$ ve $g(X)$ değerlerini arama tabloları (ing. lookup table) aracılığıyla alır ve kapı gecikmesini (Denklem 3.2) ile hesaplar. Arama tabloları 45nm Açık kaynak hücre kütüphanesi [66] simülasyonları temel alınarak oluşturulmuştur. Üretim öncesi aşamalarda hızlı gecikme hesaplamaları için, gecikme üzerindeki giriş eğimi (inp. input slope) etkisi göz ardı edilmektedir.

Üretim sonrası testleri hassas bir şekilde simüle etmek için, sadece rastgele süreç parametrelerini değil aynı zamanda giriş eğimini de dikkate alan *SLE* 'nin geliştirilmiş bir versiyonunu kullanmaktayız. Daha fazla hassasiyet sağlamak için, her kapı tipinin hem yüksekten düşüğe (ing. high to low) hem de düşüktен yükseğe (ing. low to high) gecikmeleri modellemek için iki farklı arama tablosu kullanılmaktadır [44]. Hem yüksekten düşüğe hem de düşüktен yükseğe değer kullanmanın etkisini ve doğrulamasını Bölüm 3.6 da incelenmektedir.

n_p ve n_q düğümleri arasındaki kenar $e_{p,q}$ olarak gösterilmektedir. Rastgele devre parametrelerine göre değerler atandığında, bir yola P ait gecikme değeri, n_1 ve n_k sırasıyla birincil giriş ve çıkışları temsil ederken, yoldaki kapı gecikmelerinin toplamıdır (Denklem 3.3). Kapı gecikmeleri toplanırken X örneklerine göre değerler kullanılmaktadır.

$$d_P(X) = \sum_{i=1}^k d_{n_i}(X) \quad (3.3)$$

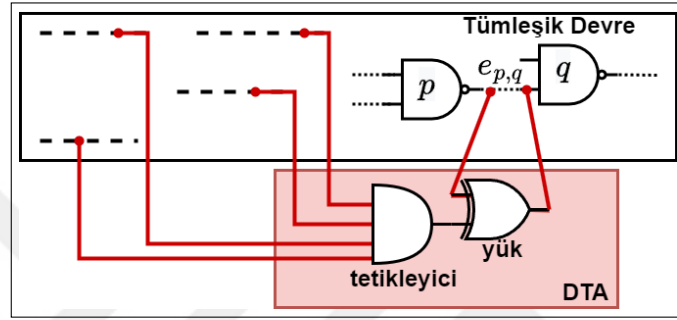
Truva atı eklenmiş yola ait gecikme değeri (Denklem 3.4), ilgili yola gecikme değerine Truva atının yük kısmına ait gecikme değerinin eklenmesiyle bulunmaktadır. Bu denklemde, Truva atının yük kısmının ilgili yola eklendiği varsayılmaktadır.

$$d_{\hat{P}}(X) = \sum_{i=1}^k d_{n_i}(X) + d_{n_{load}}(X) \quad (3.4)$$

3.4. Kullanılan Truva Atı Modeli

Farklı özelliklerine göre sınıflandırılan birçok Truva atı bulunmaktadır. Bu sınıflandırmalar fiziksel, tetikleme mekanizması ve etki özelliklerine göre ayrılmaktadır, detayları

Bölüm 1.2.2’de anlatılmaktadır. Her Truva atı tespit yöntemi, belirli Truva atı türlerine karşı farklı zayıflıklar göstermektedir. Mantıksal test tabanlı tespit yöntemleri (Bölüm 2.1.2), Truva atının boyutundan etkilenmemektedir; ancak, yan kanal analizi (Bölüm 2.1.3) tabanlı yöntemler, küçük boyutlu Truva atlarını tespit etmede başarılı olamamaktadır. Çünkü Truva atı küçük olduğunda, gecikme veya güç üzerindeki etkisi de küçülmektedir. Bu nedenle Truva atının gecikmeye olan etkisi süreç varyasyonları altında gizlenmektedir.

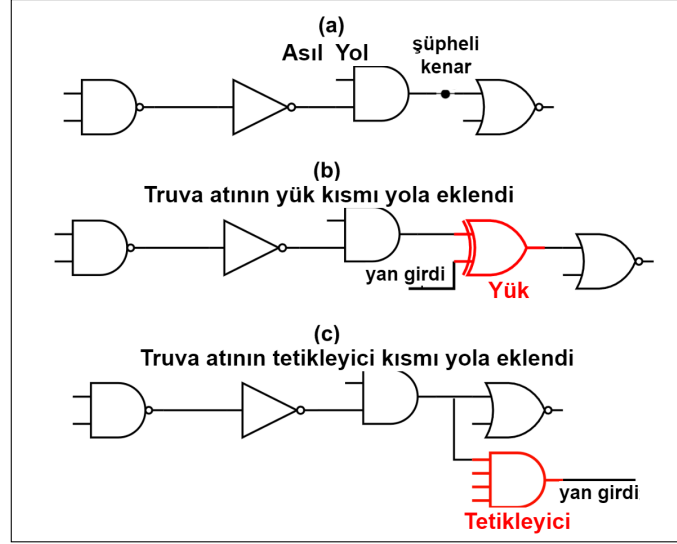


Şekil 3.4: Kullanılan Donanımsal Truva Atı Modeli.

Şekil 3.5’de, Truva atını etkinleştirmek için bir tetikleyici örneği olarak dört girişli bir VE kapısı kullanılmaktadır. Dört giriş de yüksek voltaj olan ‘1’ değerinde olduğunda, Truva atı etkinleşir ve yol üzerindeki sinyali tersine çevirmeye başlar. Çalışmamız açısından, Truva atının etkinleştirilmesinin herhangi bir önemi yoktur. Bu tipte bir Truva atını, çoğu Truva atı tespit yönteminin tespit etmesi oldukça zor olduğu için pasif olarak kabul etmekteyiz.

Şekil 3.5 (b) kısmında gösterildiği gibi, şüpheli bir kenara Truva atı yükü eklenmektedir. Şekil 3.5 (c) kısmı ise, Truva atının tetikleyici kısmının eklenmesini temsil etmektedir. Hem yük kısmının hem tetikleyici kısmının devreye bağlı olduğu durumlar için, ayrı ayrı test senaryoları oluşturulmuştur, detaylar Bölüm 5.4 de gösterilmektedir.

Yöntemi tüm devre için doğrulamak ve olabildiğince testleri zor koşullar altında denemek amacıyla, Truva atının yükünü her bir kenara ayrı ayrı eklemekte ve tespitin sağlanıp sağlanmadığını kontrol etmekteyiz. Çünkü doğrulama için rastgele kenarların seçilmesi yanıltıcı sonuçlara neden olabilmektedir.



Şekil 3.5: Truva Atının Devreye Eklenme Çeşitleri.

3.5. Notasyonlar

Tanım (Şüpheli Kenar). Bir Truva atı'nın yerleştirilebileceği bir kenar *şüpheli kenar* olarak adlandırılır.

Tanım (Şüpheli Yol: P_{susp}). Şüpheli bir kenardan geçen bir yol, o kenar için *şüpheli yol* olarak adlandırılır.

Tanım (İlişkili Yol Adayı). Tüm kapıları şüpheli bir yolla aynı ızgaralarda bulunan bir yol, bu şüpheli yol için bir *ilişkili yol adayıdır*.

Tanım (İlişkili Yol). İlişkili yol adayları arasında Truva atı tespiti için kullanılabilecek olan yola, *ilişkili yol* denir.

Tanım (Yol Çifti: $\langle P_{susp}, P_{corr} \rangle$). İlişkili yol ve buna karşılık gelen şüpheli yol bir *yol çifti* oluşturur.

Tanım (Paylaşılan Kenarların Oranı: SG). Bir çiftteki her iki yola ait kapıların bulunduğu ızgaraların oranı. İki yol ne kadar çok ortak ızgaraya sahipse, o kadar çok korelasyonlu gecikmeye sahip olma eğilimindedirler.

Tanım (Paylaşılmayan Kenarlar). Bir çiftteki iki yolda birden bulunmayan kenarlara *paylaşılmayan kenar* denir. Bu, çift içindeki yollardan yalnızca birinin o kenarı içerebileceği anlamına gelir. Bir çift için paylaşılmamış kenarlar kümesi şu şekilde gösterilir:

$$E(P_{susp}, P_{corr}).$$

Tanım (Kenar Kapsamı). Yol çifti bulunan bir kenar *kapsanabilir* kenar olarak adlandırılır. Herhangi bir kenar için kullanılan yöntem herhangi bir yol çifti bulamazsa,

bu kenar *kapsanamaz* olarak adlandırılır. Üretim sonrası sonuçlar bir kenarın kapsama durumunu doğrularsa, bu kenar *kapsanan* kenar olarak adlandırılır. Üretim sonrası testlerden sonra kapsanan toplam kenarların oranı *kenar kapsamı* olarak adlandırılır. Bu değer tespit olasılığına denk gelmektedir.

Tanım (Gecikme Oranı: R or $\hat{R}$). Bir yol çiftinin gecikme oranı, çiftteki yolların gecikme değerlerinin bölünmesiyle hesaplanır. Şüpheli yolda Truva atı yoksa, oran R ile gösterilir, aksi takdirde eklenen bir Truva atı varsa, $\hat{R}$ ile gösterilir.

Tanım (Dağılım Oran Farkı: $DD(R, \hat{R})$). Eğer R ve $\hat{R}$ gecikme oranı dağılımları birbirlerinden standart sapmalarının 1,5 katından daha fazla ayrılıyorsa, dağılım farkı pozitifdir. Aksi takdirde negatiftir. Bu metrik iki dağılımın birbirinden ne kadar ayrıldığını anlamak için kullanılır. Çünkü 1, 5σ 'dan daha fazla ayrılan dağılımların bimodal olduğu tespit edilebilir ve bu durum Bölüm 4.1.5 da açıklanmıştır. Dağılım farkı aşağıda gösterildiği gibi hesaplanır:

$$DD(R, \hat{R}) = (\mu_{\hat{R}} - 1, 5\sigma_{\hat{R}}) - (\mu_R + 1, 5\sigma_R) \quad (3.5)$$

3.6. Hassas Stokastik Lojik Efor Yöntemi

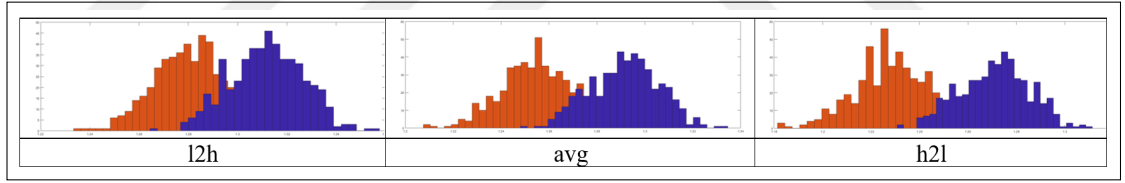
Üretim sonrası testlerde Hassas Stokastik Lojik Efor (*HSLE*) (ing. precise SLE) [44] modelini kullanmaktayız. Bu modelin yol gecikme ölçümlerini, SPICE ölçümleri ile kıyaslayarak, doğruluğunu 3.9'da gösterilmiştir. *HSLE* modelinde mümkün olduğunca gerçeğe yakın hassasiyette sonuçlar elde etmek istiyoruz. Bu nedenle, hem $L2H$ hem de $H2L$ yol gecikme ölçümlerini dikkate almaktayız.

- $L2H$ (ing. low to high): İlgili yola ait çıktının gerilim değeri düşük gerilim olan sıfırdan, yüksek gerilim olan bir değerine çıkarken ölçülen yol gecikmesidir.
- $H2L$ (ing. high to low): İlgili yola ait çıktının gerilim değeri yüksek gerilim olan bir değerinden, düşük gerilim olan sıfır değerine inerken ölçülen yol gecikmesidir.

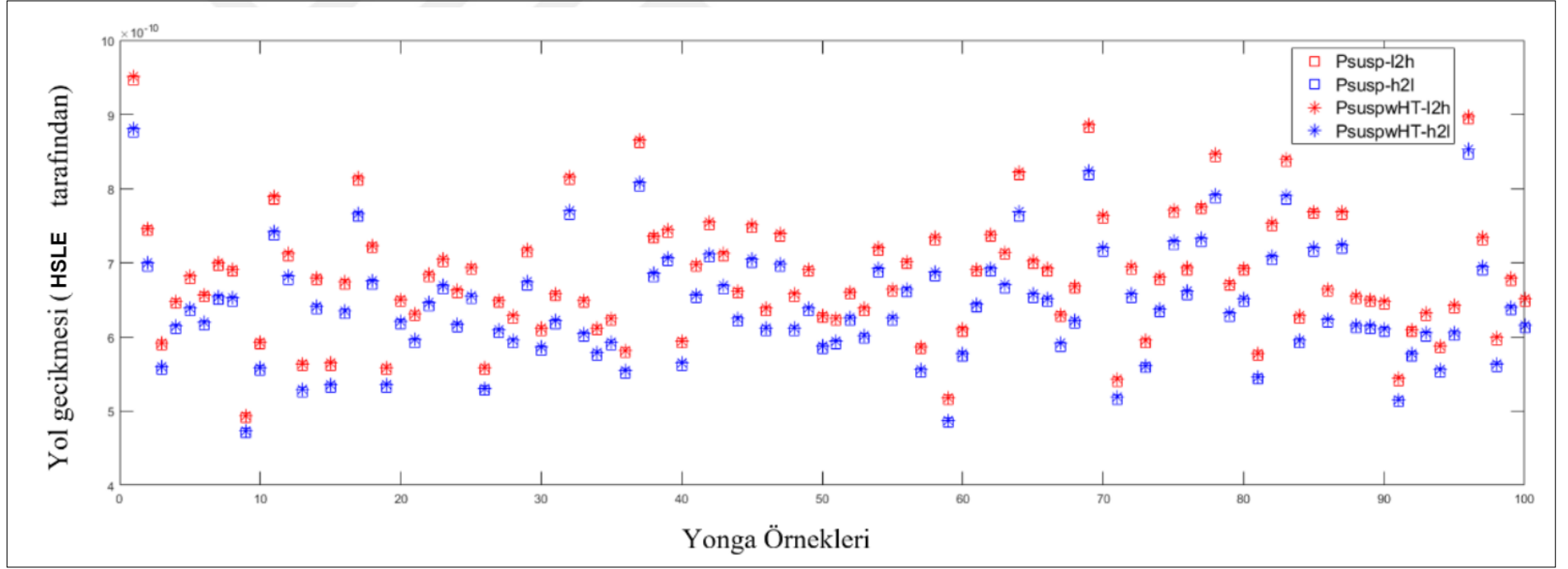
Şekil 3.7'de, $Psusp$ yolun Truva atsız halini, $\hat{Psusp}$ ise Truva atının tetikleyici kısmının yola eklendiği halini ifade etmektedir. Örneklerde yol gecikme ölçümleri gösterilmektedir. Şekilde yıldız (*) sembolüyle gösterilen gecikme değerleri $\hat{Psusp}$ yoluna, kare sembolüyle gösterilenler $Psusp$ yoluna aittir. Örneklerin çoğunda, $L2H$ ile $H2L$ arasındaki fark, $L2H$ ile $\hat{L2H}$ ve $H2L$ ile $\hat{H2L}$ arasındaki farktan daha büyüktür. Buradan

ıkarılan sonu, yol gecikmesini doėru bir ekilde lmezsek, Truva atının etkisinin gizlenebileceėidir.

Yaptıėımız lmler sonucunda, tetikleyici kısmının yol gecikmesine etkisini tespit ederken, ortalama yol gecikmesini oluřturan $L2H$ ve $H2L$ yol gecikmelerini ayrı ayrı dikkate almanın nemini fark ettik. řekil 3.6’da yol gecikmesi oranlarına ait histogramlar gsterilmektedir; turuncu renkle gsterilenler Truva atsız yol gecikmesi oranı ($Ratio$), mavi renkle gsterilenler Truva atlı yol gecikmesi oranı ($\hat{Ratio}$)’na aittir. Bu rnekte, $L2H$ ve $H2L$ yerine, bu iki gecikme deėerlerinin ortalaması olan avg deėerini kullanıldığında, Truva atının tespiti olduka zorlařmaktadır. Bunun yerine ayrı ayrı $L2H$ ve $H2L$ yol gecikmeleri kullanıldığında, rnekteki $H2L$ gecikme oranına ait histogramdan, Truva atı tespit edilebilmektedir. Sonu olarak, $HSLE$ ynteminde $L2H$ ve $H2L$ gecikme deėerleri ayrı ayrı hesaplanmakta ve kullanılmaktadırlar. Bu sayede, yntemimizin yonga-ii ve yongalar arası varyasyonların etkilerini ortadan kaldırabildiėini gstermekteyiz. $L2H$ ve $H2L$ kullanarak elde ettiėimiz Truva atı tespit oranına dair sonular Blm 5.4 da gsterilmektedir.



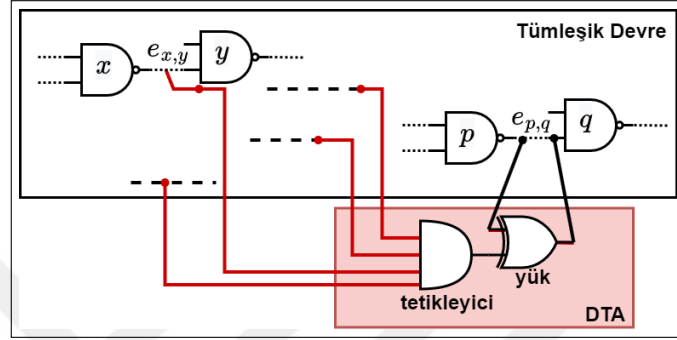
řekil 3.6: $L2H$, $H2L$ ve ortalama gecikme deėerleri kullanıldığında elde edilen gecikme oran daėılımları.



Şekil 3.7: L2H ve H2L Gecikme Ölçümlerinin Gösterimi.

3.7. Yol Gecikmesi Ölçümlerinin Doğrulanması

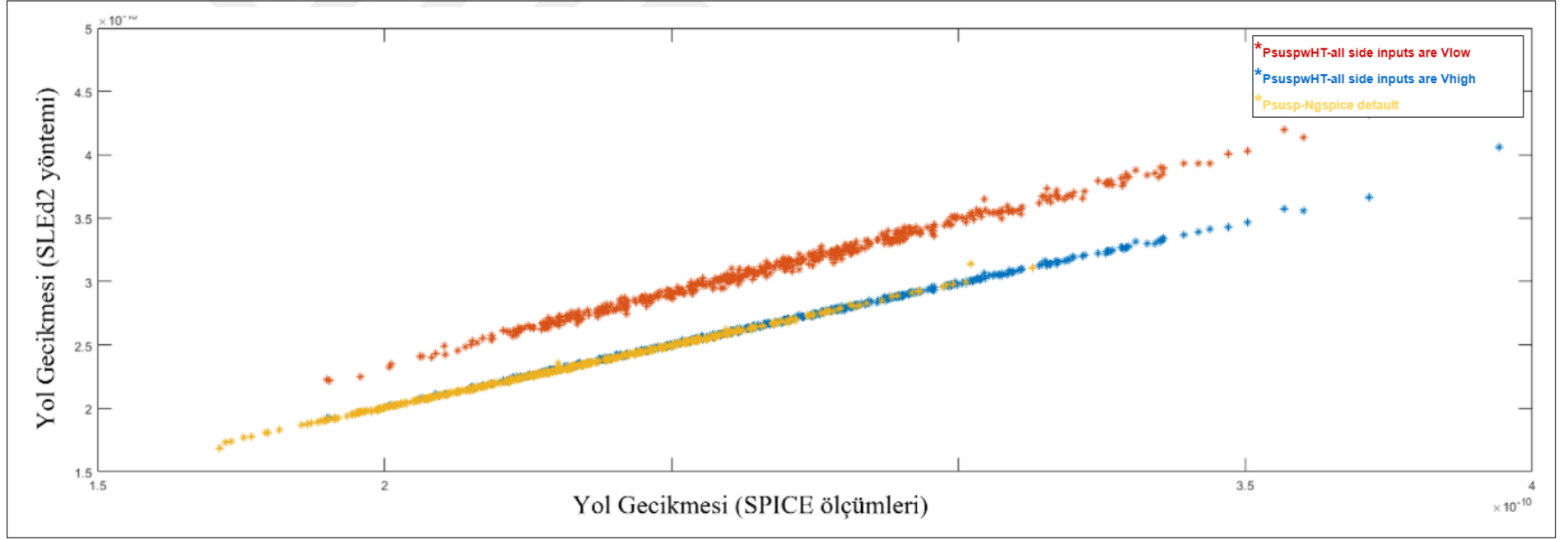
SPICE [23] en ünlü transistör seviyesi devre simülatörüdür. Bugün, SPICE elektronik tasarım otomasyonu (ing. EDA) endüstri standardı haline gelmiş olup, büyük şirketler hala kendi simülatörlerini geliştirmeye devam etmektedir. Yol gecikmesi için kullandığımız *HSLE* yönteminin doğruluğunu kontrol için SPICE kullanılmaktadır.



Şekil 3.8: Truva Atının Tetikleyici Kısmı olan Yoldışı Parametrelerin Etkisi.

Yoldışı transistörler, Şekil 3.8’de kırmızı renkle gösteren kenarlardır. Devrede ölçümler sırasında oluşan küçük geçiş farkları (ing. transition) yoldışı (ing. off-path) transistörlerden kaynaklanmaktadır. Örneğin, seri transistör çiftinden biri zaten açık iken geçiş, her iki transistörü aynı anda açmaktan daha hızlı olacaktır. Bu farklar, gecikme ölçümünün doğruluğunu ve dolayısıyla Truva atlarına karşı duyarlılığı etkileyebilirler.

Şekil 3.9’da, Truva atının tetikleyici kısmı olan VE kapısının yan girişlerinin sıfır (ing. Vlow) ve bir (ing. Vhigh) olduğu durumlarda ne olduğu incelenmiştir. Kırmızı renk, Truva atının tetikleyici kısmının tüm yan girişlerinin sıfır olarak atandığını, mavi renk ise tetikleyici kısmının tüm yan girişlerinin bir olarak atandığını temsil etmektedir. SPICE, yoldışı transistörlere bir verildiğini varsayılarak modellenmiştir. Bu nedenle, bizimde tüm yan girişlerine bir değeri atanmış olan, mavi renkli sonuçları kullanmamız daha doğru ölçümler sağlamaktadır. Aynı zamanda, Truva atının tetikleyici kısmının yan girişlerinin bir olması, Truva atının aktif olduğu anlamına gelmektedir. Bu da, yöntemimizin Truva atını hem aktifken hem de pasifken tespit edebileceği anlamı taşımaktadır.

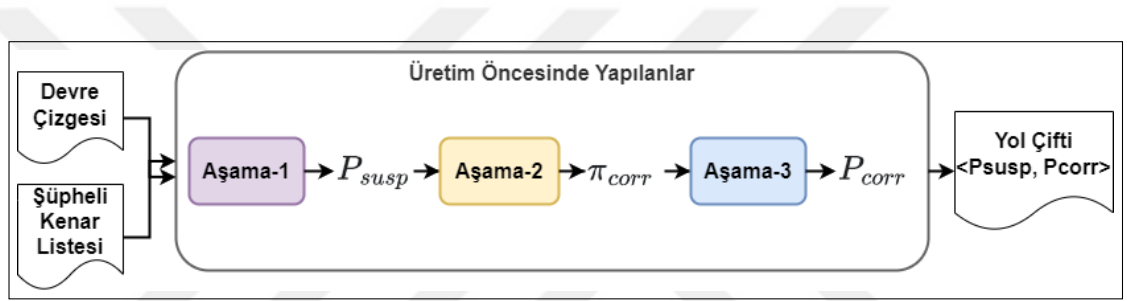


Şekil 3.9: Simulasyon ve SPICE sonuçlarının Karşılaştırılması.

4. Konumsal Korelasyonlardan Yararlanarak Yol Gecikmesi ile Truva Atı Tespiti

Çalışmamız üretim öncesi (ing.pre-manufacturing) ve sonrası (ing.post-manufacturing) olmak üzere iki ana bölümden oluşmaktadır. İlk bölümde üretim öncesine ait yöntemleri inceleyeceğiz.

Üretim öncesi için geliştirdiğimiz yöntemlerimizin tamamında kullandığımız üç aşamalı bir yapı mevcuttur. Bu yapı Şekil 4.1’de gösterilmektedir. Her bir yöntem, üretim öncesi yapılanların sonunda yol çifti setini elde etmektedir.



Şekil 4.1: Üretim öncesinde yapılması gereken aşamaların gösterimi.

Süreç varyasyonlarında doğuştan bulunan konumsal korelasyonlar, lojik kapılar birbirine ne kadar yakınsa, süreç varyasyonlarından o kadar benzer şekilde etkilendiklerini ifade etmektedir. Süreç varyasyonu ile ilgili detaylar Bölüm 1.4’de anlatılmaktadır. Üretim öncesinde amacımız, konumsal korelasyonlardan yararlanarak tespit edilmiş yol çiftlerini elde etmektir. Amacımıza ulaşmak için takip edeceğimiz adımlar üç aşamada anlatılmaktadır ve devredeki her bir şüpheli kenar için gerçekleştirilmektedir.

Bunun için üç farklı yöntem geliştirilmiştir. Yöntemlere ait detaylar, yöntemlerin birbirlerine göre avantajları ve kıyaslamaları ayrı altbaşlıklarda incelenmektedir. Her bir yöntem aşağıda bahsedilen aşamalardan oluşmaktadır. İlk üç aşama üretim öncesine aittir. Ancak son aşama üretim sonrasında, üretilen yongalar üzerine uygulanmaktadır.

- Aşama-1: Her şüpheli kenar için bir adet şüpheli yolun elde edilmesi,
- Aşama-2: Her bir şüpheli yol için onunla ilişkili yol adaylarının elde edilmesi,
- Aşama-3: Bu adaylar arasından bir adet ilişkili yolun seçilmesi şeklinde aşı-

maların temel hedefleri özetlenebilir.

- Aşama-4: Yol ikilisi kümesindeki her bir yol için, yol gecikmelerinin ölçülmesi ve birbirine bölünmesini içermektedir. Oranlamanın amacı, varyasyonların etkisinden kurtulmaktır. Truva atının tespiti için; yol çiftine ait gecikme değerlerinin birbirlerine bölünmesi sonucunda sabit bir değer elde edilip edilmediğinin analizi yapılır.

Konumsal korelasyona dayalı Truva atı tespiti, herhangi bir kenara eklenmiş Truva atını yol çiftlerinin gecikme oranıyla tespit etmeyi amaçlamaktadır. Ancak, devredeki her bir kenar için ilgili bir yol çifti bulmak zordur.

Kapılar birbirine ne kadar yakınsa, varyasyonlardan o kadar benzer şekilde etkilendiklerini görmüştük (Bölüm 3.2). Eğer aynı ızgaralarda (Şekil 4.4) yer alan kapıları olan iki yol çıkarabilirsek, bu yolların gecikmelerinin varyasyonlardan benzer şekilde etkilenecektir. Sonuç olarak, bu iki yola ait gecikme değerleri oranlandığında, varyasyonlardan çok fazla etkilenmeyen stabil bir oran elde edilmektedir. Buna ek olarak, aynı ızgaralarda yer alan tüm çiftlerin yol gecikmelerinin, varyasyondan tamamen aynı şekilde etkileneceğinin kesinliği bulunmamaktadır.

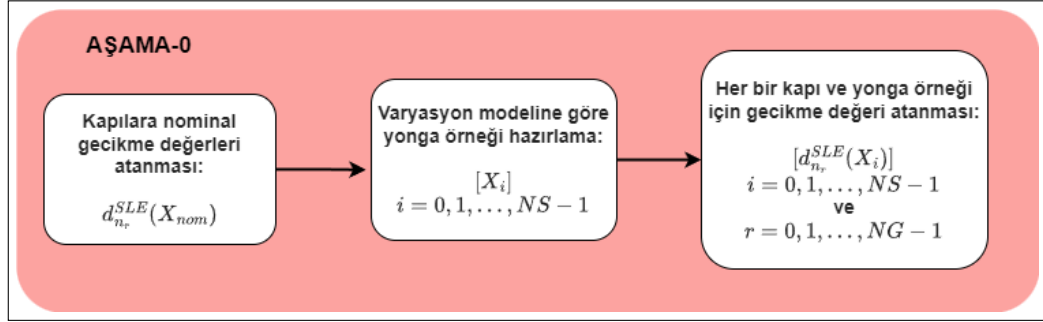
4.1. Korelasyon Katsayısına Dayalı Truva Tespit Yöntemi (KKY)

Korelasyon Katsayısına Dayalı Yöntem (KKY) [75] , varyasyonlardan benzer şekilde etkilenen bileşenleri bulmayı ve bunları oranlayarak, varyasyonların etkisini ortadan kaldırmayı amaçlamaktadır. Tüm yöntemlerde olduğu gibi, bahsedilen dört aşama ile anlatılacaktır.

4.1.1. Aşama-0: Kapı Gecikmesi Hesabı

Truva atı tespit yöntemini devrede uygulamadan önce, kapı gecikmelerinin nominal olarak hesaplanması gerekmektedir. Gecikme hesaplamaları için Bölüm 3.3'te açıklanan SLE yöntemini kullanmaktayız. İlk olarak, devredeki rastgele parametreler (L_{eff} and V_t) için Bölüm 3.2'te anlatıldığı şekilde değerler atanmaktadır. İkincisi, her biri devrenin bir yonga kopyasını temsil eden örnekler üretilir. Sonrasında, oluşturulan örnekler X_i ve

3.2 kullanılarak, her bir kapı için örnek sayısı kadar gecikme değeri hesaplanır (Denklem 4.1).



Şekil 4.2: Kapı Gecikmesi Hesabı için Yapılanlar.

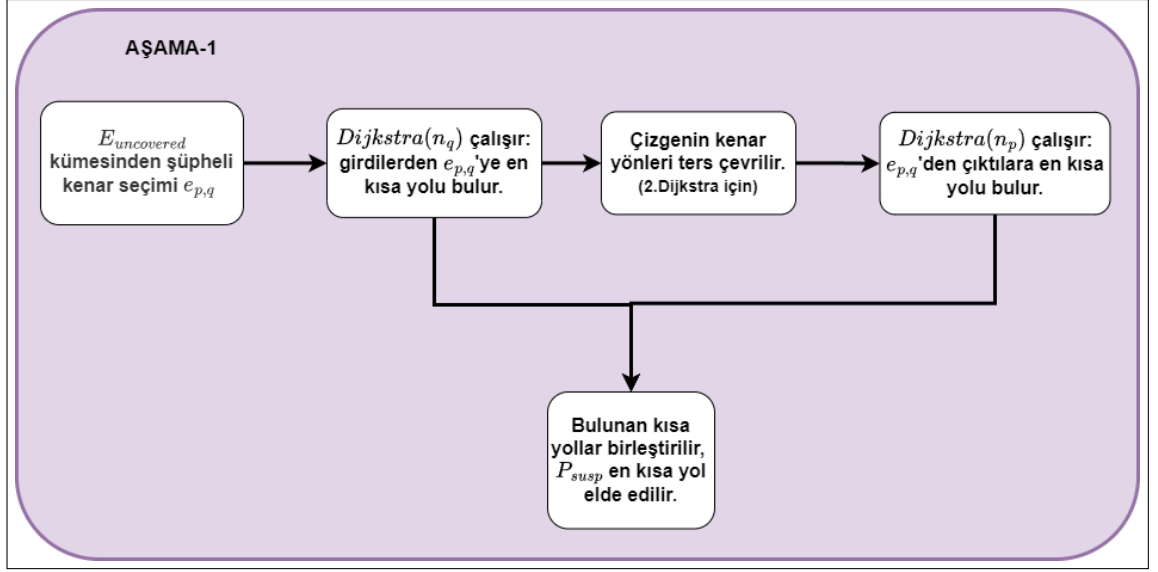
$$[d_{n_r}^{SLE}(X_0), d_{n_r}^{SLE}(X_1), \dots, d_{n_r}^{SLE}(X_{NS-1})] \quad (4.1)$$

Şekil 4.2 NS değeri, Monte Carlo yöntemiyle çekilen örneklerin sayısını ifade etmektedir. Bu gecikme değerleri, Aşama-2 ve Aşama-3 tarafından kullanılmaktadır. Daha hızlı hesaplamalar için, giriş eğiminin gecikme üzerindeki etkisi göz ardı edilmekte ve yüksekten düşüğe ve düşükten yükseğe gecikmeler yerine yalnızca ortalama gecikme dikkate alınmaktadır. Bu, tek seferlik bir işlemdir ve doğrusal bir model kullanıldığı için tüm gecikmeler çok hızlı bir şekilde hesaplanmaktadır. Bu işlemlerin tamamı, tespit yöntemleri çalışmaya başlamadan önce sadece bir kez yapılmaktadır.

4.1.2. Aşama-1: Şüpheli Yol (P_{susp}) Elde Etme

Mantıksal temelli yöntemler kullanarak veya devrenin kritik bölümlerini dikkate alarak, devredeki şüpheli kenarların sayısı önemli ölçüde azaltılabilmektedir. Ancak, bu makalede yöntemin gerçek performansını görmek için, birincil girişler ve çıkışlar hariç tüm kenarların şüpheli olarak işaretlendiğini varsaymaktayız. Çünkü bir birincil giriş veya çıkışa bağlı bir Truva atı, kolayca pin tabanlı ölçümlerle tespit edilebilmektedir.

Kısa yollar için tespit olasılığı, Truva atının yol gecikmesi üzerindeki göreceli etkisi nedeniyle daha fazladır [17]. Şüpheli bir kenardan geçen en kısa yolu bulmak için, önce n_q düğümü için Dijkstra algoritması çağrılarak, birincil çıkışa en kısa yol bulunur. Sonra, grafin kenarları tersine çevrilerek n_p düğümü için Dijkstra çağrılır, bu da birincil



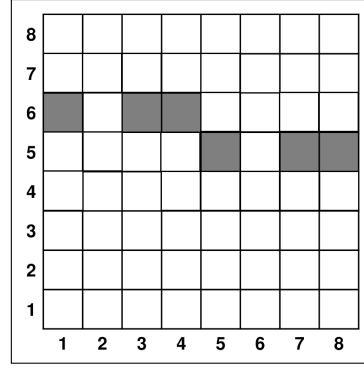
Şekil 4.3: Şüpheli Yol Seçimi.

girişe en kısa yolu verir. Bulunan iki yolun şüpheli kenarla birleşimi, $e_{p,q}$ kenarından geçen en kısa yolu verir.

Algoritma bir şüpheli yol bulduktan sonra, bir çift oluşturmak için ilişkili yolu bulmak üzere sonraki aşamalara geçer. $E_{uncovered}$ kümesinde şüpheli kenarlar bulunmamaktadır. Başlangıçta $E_{uncovered}$ kümesine, devredeki tüm kenarlar şüpheli olarak eklenmektedir. Aşama-1'in devredeki her kenar için çalıştırıldığı düşünülürse, toplam maliyet her kenar için iki Dijkstra çağrısı kadardır. Ancak, Aşama 4.1.4'da açıklandığı gibi, bir yol çifti birden fazla şüpheli kenar için kullanılabilir. Yani $E_{uncovered}$ boyutu her yinelemede birden fazla sayıda azaltılabilmektedir. Bunun sonucunda, uygulama kısmında Aşama-1'in devredeki şüpheli kenar sayısından daha az çalıştığını ifade edebiliriz.

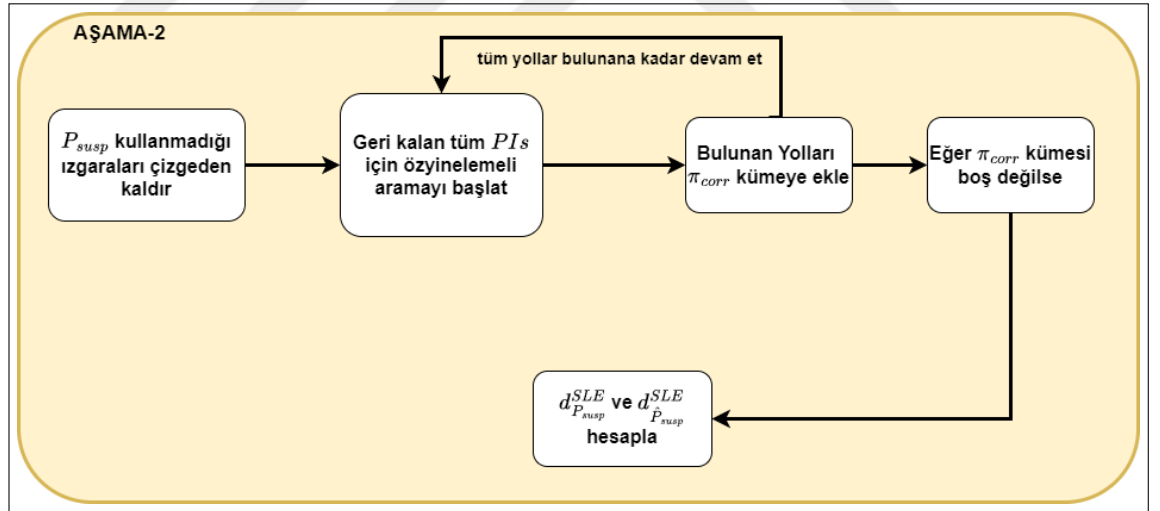
4.1.3. Aşama-2: İlişkili Yol Adayları Kümesi (π_{corr}) Oluşturma

Şüpheli yol ile ilişkili olabilecek tüm olası yol adaylarının çıkarılması aşamasıdır. Bunun için konumsal korelasyon bilgisi kullanılmaktadır. Konumsal korelasyonlar nedeniyle, başka bir yol ile çok yakın konumlardaki lojik kapılara sahip olan bir yol, aynı zamanda korelasyonlu yol gecikmelerine de sahip olmaktadır. Şüpheli bir yol için çok yakın konumlu başka bir yol bulunabilirse, bu iki yolun yol gecikmelerinin oranının, varyasyon bileşenini ortadan kaldırabileceği anlamına gelmektedir. Bu durumda, ilgili yol üzerine



Şekil 4.4: Tüm devre yerleşiminin ızgaralara bölünmesi.

Truva atı eklenmesi gibi herhangi bir değişiklik, yol gecikme oranındaki sapma tespit edilerek kolayca ortaya çıkarılabilecektir. Şüpheli bir yola ait ilişkili yol adaylarını bulmak için devre ızgaralara bölünür (Şekil 4.4). Sonrasında, kapıları aynı ızgarada bulunan tüm yollar özyinelemeli (ing.recursive) olarak çıkarılır ve bu yollar, ilişkili yol adayları kümesinde toplanır. Sonunda, her şüpheli yola karşılık gelen bir ilişkili yol adayları kümesi elde edilir.



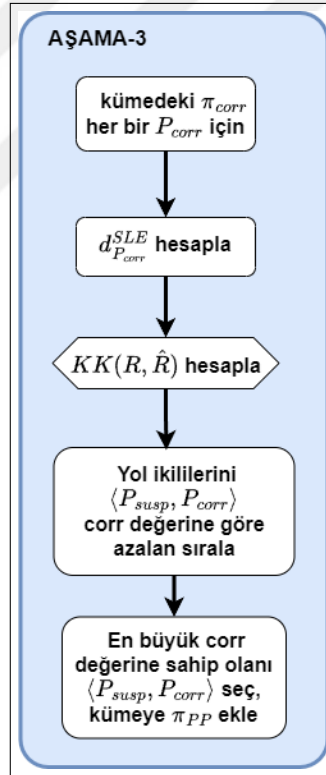
Şekil 4.5: İlişkili Yol Adayları Kümesi Oluşturma.

Truva atının tespti edilebilmesi için her iki yol tarafından paylaşılmayan bir kenara eklenmiş olması gerekmektedir. Aksi takdirde, eklenen Truva atının gecikmesi her iki yolu da eşit şekilde etkileyecek ve oranla tespit imkansız hale gelecektir. İlişkili yol seçerken atlanmaması gereken önemli bir nokta da budur. Bu koşul, yol çiftleri seçimini zorlaştırmaktadır. Şüpheli bir kenarda Truva atı olup olmadığının kontrolü için, şüpheli

kenarın sadece yollardan birinde bulunması ve yol çiftlerinin yüksek korelasyona sahip olmaları gerekmektedir.

4.1.4. Aşama-3: Yol İkili ($\langle P_{susp}, P_{corr} \rangle$) Seçimi

Öncelikle tüm ilişkili yol adaylarının yol gecikmeleri yonga örnekleri (ing. samples) için hesaplanır. Ardından, ilişkili yol adayları kümesindeki her bir yol, ilgili şüpheli yol ile eşlenerek yol çifti oluşturulur. Her bir yol çifti için, bu yol gecikmelerine dayalı olarak korelasyon katsayısı (KK) hesaplanır. Aday yollar arasından şüpheli yol ile en çok ilişkilendirilen yolu seçmek için, en yüksek korelasyon katsayısına sahip olan ilişkili yol adayı aranmaktadır. En iyi korelasyon katsayısına sahip yol adayı, e kenarına ait şüpheli yol için seçilmiş ilişkili yol (P_{corr}^e) olarak belirlenir. Sonunda, ilgili şüpheli yol için yol çifti elde edilmiş olur.



Şekil 4.6: İlişkili Yol Adayları Kümesi Oluşturma.

4.1.5. Aşama-4: Üretim Sonrası Truva Atı Tespiti

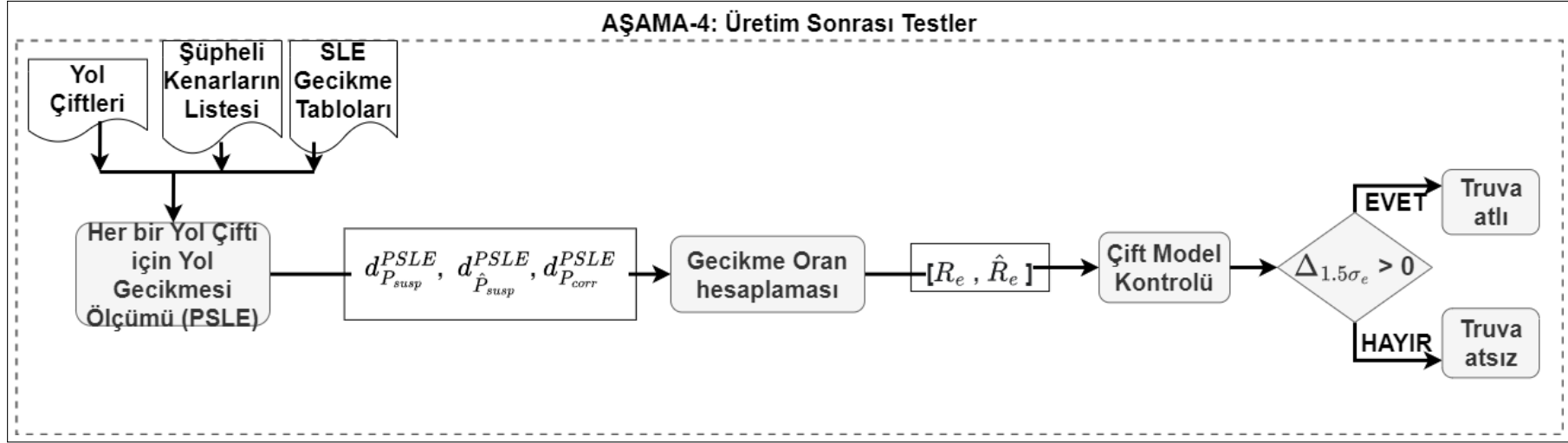
Üretim sonrası test aşamasından sorumlu kısımdır, Şekil 4.7’de detayları gösterilmektedir. Şüpheli yolun ve aday gösterilen ilişkili yolun, yol gecikmeleri hassas stokastik lojik efor *PSLE* yöntemiyle [44] ölçülmektedir. Ardından, ilgili şüpheli yol ve ilişkili yol adayından oluşan her bir yol çifti için gecikme oranı hesaplanır. Gecikme oranı, yol çiftindeki iki yola ait gecikme değerlerinin oranlanmasıyla hesaplanmaktadır (Denklem 4.2). Örnek bir şüpheli kenar e için gecikme oranının hesaplanması denklem (4.2) ile gösterilmektedir. Burada d yol gecikmesini ifade etmektedir. Burada gecikme oranı hesabının amacı, varyasyon etkisini ortadan kaldırmaktır. Böylece herhangi bir Truva atı varlığı, gecikme oranında oluşan değişimden tespit edilebilmektedir.

$$R_e = \frac{d(P_{susp}^e)}{d(P_{corr}^e)} \quad (4.2)$$

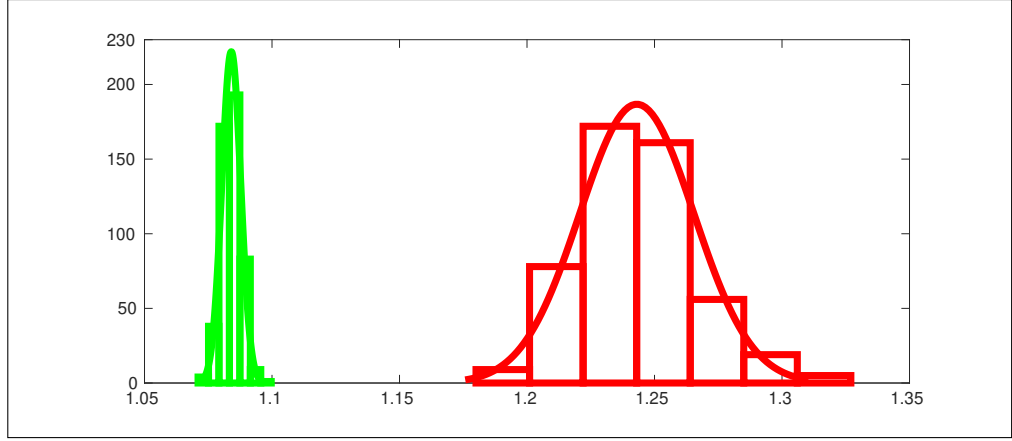
$$\Delta_{1,5\sigma_e} = (\mu_{\hat{R}_e} - 1, 5\hat{\sigma}_{\hat{R}_e}) - (\mu_{R_e} - 1, 5\sigma_{R_e}) \quad (4.3)$$

Yol çifti, eğer $1, 5\sigma$ pozitif farka sahipse iki dağılım birbirinden ayrılmaktadır. Bir e kenarı için Truva atı içermeyen ve Truva atı yerleştirilmiş örneklerin gecikme oranları arasındaki $1, 5\sigma$ farkının hesaplanması denklem (4.3) ile gösterilmektedir. Burada $\Delta_{1,5\sigma_e}$, e kenarı için $1, 5\sigma$ farkını, μ_{R_e} ve σ_{R_e} Truva atı içermeyen örnekler için gecikme oranlarının ortalaması ve standart sapmasını, $\mu_{\hat{R}_e}$ ve $\hat{\sigma}_{\hat{R}_e}$ Truva atı eklenmiş örnekler için gecikme oranlarının ortalaması ve standart sapmasını ifade etmektedir.

Eğer Dağılım Farklarına ait $1, 5\sigma$ değeri pozitif değerde ise, gecikme oranlarının dağılımları çift modelden oluşmaktadır. Bunun anlamı da, Truva atı içermeyen ve Truva atı eklenmiş örnekler için, altın model gerektirmeden Truva atı tespitinde başarılı olacak demektir. Eğer $1, 5\sigma$ değeri pozitif değilse, gecikme oranlarının dağılımları tek modelden oluşmaktadır. Yani bu yol çifti truva atı tespiti için uygun değil demektir. Bunun sonucunda, ilgili kenarlar hala $E_{uncovered}$ kümesinde bulunmaya devam ederler. İlgili şüpheli kenara ait, Truva atsız ve Truva atlı örneklerin gecikme oran dağılımlarının $1, 5\sigma$ değeri pozitif ise, bu kenar *kapsanmış* (ing.covered) olarak adlandırılır. Bu nedenle, eğer Aşama-4’de kenar tespit edilebiliyorsa, bu KKY yönteminin ilgili kenar üzerinde Truva atı tespiti yapabilen, uygun yol çiftinin seçildiği anlamına gelmektedir.



Şekil 4.7: Üretim Sonrasında Truva Atı Tespiti.



Şekil 4.8: DTA içermeyen (yeşil) ve DTA eklenmiş (kırmızı) Gecikme oran Dağılımlarının Gösterimi.

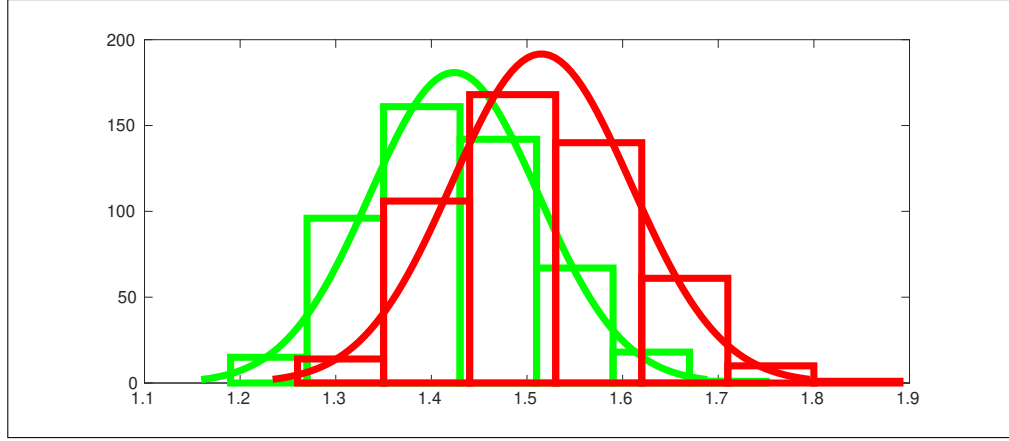
Bu algoritmanın en çok zaman alan bölümü Aşama-3'e aittir. Milyonlarca kapaıya sahip mevcut tümleşik devreler göz önünde bulundurulduğunda, ve devredeki tüm kenarlar şüpheli olarak değerlendirmeye alındığında çok sayıda ilişkili yol adayı bulunabilecektir. Bu nedenle, Aşama-3'de çok fazla sayıda yol gecikmesi ve korelasyon katsayısı hesaplanması gerekmektedir.

Örneğin, c1908 kıyaslama devresinden kapsanabilir bir kenar için gecikme oranı dağılımları Şekil 4.8'da gösterilmektedir. Bu kenar için, Truva atı yerleştirilmiş örnekler altın bir modele ihtiyaç duyulmadan kolayca ayrılabilir. Bu nedenle, şüpheli kenarın yöntem tarafından güvence altına alındığı veya kapsandığı söylenir.

Şekil 4.9 c1908'den başka bir kenara ait gecikme oranı dağılımlarını göstermektedir. Bu kenar yöntem tarafından kapsanamaz çünkü Truva atı içermeyen ve Truva atı yerleştirilmiş örneklerin gecikme oranları birbirine karışmıştır ve negatif $\Delta_{1,5\sigma_e}$ ile sonuçlanmaktadır. Buradaki gibi renklendirilmedikleri takdirde iki dağılımın birbirinden ayrılması mümkün değildir.

4.2. Paylaşılan Izgaraların Oranına göre Truva Atı Tespit Yöntemi (PIY)

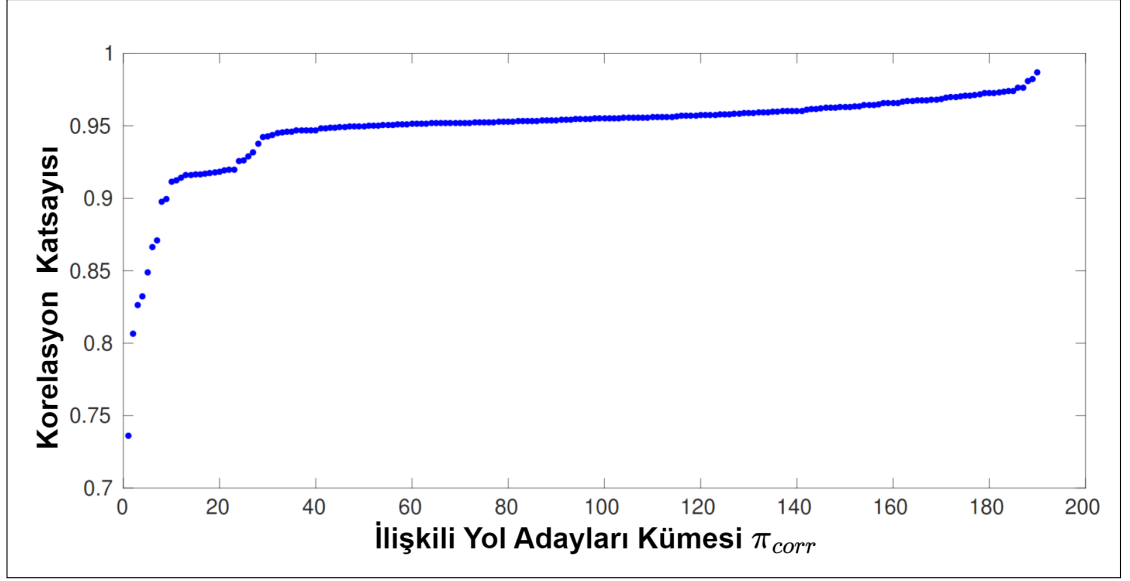
Bu bölümde, en iyi ilişkili yol adayını seçmek için daha hızlı çalışan bir yöntem önermekteyiz. Bu yöntem de dört aşamadan oluşmaktadır. Aşama-0 4.1.1, Aşama-1 4.1.2, Aşama-2 4.1.3 önceki yöntem olan KKY ile aynıdır. Aşama-3 için oldukça hızla çalışan



Şekil 4.9: DTA içermeyen (yeşil) ve DTA eklenmiş (kırmızı) Gecikme oran Dağılımlarının Gösterimi.

ve gecikme hesapları yapmaktan bizi kurturan yeni bir yaklaşım önerilmektedir. Aşama-4 de Bölüm 4.1.5’deki ile aynıdır.

Bölüm 4.1’de yöntem ile ilgili asıl problemi detaylandıralım. Konumsal korelasyon nedeniyle, tüm adayların (örneğimiz için Şekil 4.4 de yalnızca koyu gölgeli ızgaralarda bulunanların) iyi bir korelasyona ve dolayısıyla iyi bir korelasyon katsayısına sahip olması beklenir. Böyle bir durumda, her kenar (veya şüpheli yol) için tüm adayları sıralamak ve ardından hepsi için yol gecikmelerini ve korelasyon katsayılarını hesaplamak yerine sadece bir adet ilişkili yol adayı seçmek, ilişkili bir yol çifti elde etmek için yeterli olacaktır. Ancak ilişkili yol adaylarını incelediğimizde durumun böyle olmadığını görüyoruz. Sadece bir şüpheli yola karşılık gelen tüm ilişkili yol adayları için farklı korelasyon katsayıları Şekil 4.10’da örnek olarak gösterilmiştir. Adaylar artan katsayı değerlerine göre sıralanmıştır. Bu örnek durum için, bazı adayların korelasyon katsayıları 0,75’e kadar kötü değerlerde olabilir. Unutulmamalıdır ki, deneysel sonuçlar bize varyasyonların etkisini ortadan kaldırabilmek ve Truva atlarının varlığını ortaya çıkarmak için korelasyon katsayısının 0,95 ve üzeri gibi 1’e çok yakın bir değere sahip olması gerektiğini göstermektedir. Eğer Truva atı sadece bir adet lojik kapı kadar küçükse, 0,95’lik bir korelasyon katsayısı bile tespit için yeterli olmayabilir. Bu durum, her bir ilişkili yol adayı için korelasyon katsayısının hesaplanmasını ve en büyük katsayıya sahip olanın çiftin ilişkili yolu olarak belirlenmesini gerektirmektedir.



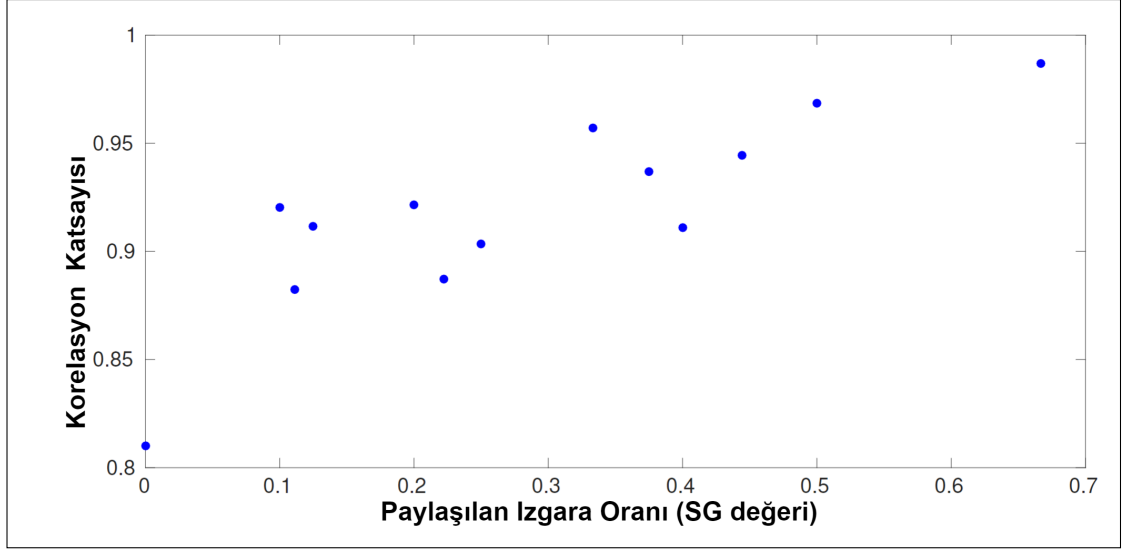
Şekil 4.10: İlişkili yol adaylarının korelasyon katsayılarının gösterimi.

4.2.1. Yol İkilişi ($\langle P_{susp}, P_{corr} \rangle$) Seçimi

Detaylı incelemeler sonucunda, şüpheli yolla aynı ızgaralarda bile bulunmanın onunla yüksek korelasyonlu olmak anlamına gelmediği gerçeğini ortaya çıkarmaktadır. Çünkü ortak ızgaraların sayısı, yol çiftinin geçtiği tüm ızgaraların sayısına göre oldukça düşük olabilmektedir. Bu da bize sadece korelasyon katsayısına bakarak en uygun ilişkili yolu seçmenin zorluğunu göstermektedir. Genelleme kaybı olmadan, şüpheli yolun farklı ızgaralara dağıtılmış kapılara n_s sahip olduğunu ve bu şüpheli yol için bir ilişkili yol adayının tüm kapılarının n_c farklı ızgaralarda bulunduğunu varsayalım; burada n_s ve n_c ızgaralarının birleşme ve kesişme sayısı sırasıyla n_U ve n_I ile gösterilmektedir. Eğer n_I değeri, n_U 'tan çok daha küçükse, bu iki yol arasında ortaya çıkan ilişki varyasyon etkisini ortadan kaldıracak kadar iyi olmayacaktır. Biz n_I/n_U oranını *shared grids ratio* (SG) olarak adlandırıyoruz. Bir yol çifti $\langle P_{susp}, P_{corr}^i \rangle$ için paylaşılan ızgara oranı (4.4)'de gösterildiği gibi hesaplanabilmektedir. i şüpheli yola ait ilişkili yol adayının indeksini gösterir. İki yol arasındaki korelasyon, ortak ızgara oranı arttıkça artma eğilimindedir.

$$SG^i = \frac{\langle P_{susp}, P_{corr}^i \rangle \text{ yol çiftinin paylaşımlı ızgara sayısı}}{P_{susp} \cup P_{corr}^i \text{ tüm ızgaraların sayısı}} \quad (4.4)$$

KKY yöntemindeki Aşama-3'ün hızlandırılması için, en büyük korelasyon katsayısına



Şekil 4.11: İlişkili yol adayları seçiminde kullanılan değerlerin karşılaştırması

sahip olanı seçmek yerine (4.4) tarafından hesaplanan en büyük SG oranına sahip ilişkili yol adayının seçilmesini önermekteyiz. PIY yönteminin KKY yönteminden farkı sadece Aşama-3 de ilişkili yol seçimi kısmındadır. Bunun nedeni de Aşama-3'ün dar boğaz olmasıydı. PIY sayesinde kazanılan hız ve tespit oranı detayları Bölüm 5.2'de gösterilmektedir.

Şekil 4.11'de örnek bir şüpheli yol için 15 adet ilişkili yol adayı seçilmiştir. Her bir yol adayı için her iki yöntemde de, Aşama-3 de kullanılan değerlendirme yapısı uygulanmıştır. KKY yönteminde korelasyon katsayısı ve PIY yönteminde SG değeri hesaplanmış ve sırasıyla y,x eksenleriyle gösterilmiştir. Örnek bir şüpheli yol için Şekil 4.11, SG değerinde bazı dalgalanmalar olmasına rağmen, aynı yolun korelasyon katsayısının nasıl yükselme eğiliminde olduğunu göstermektedir.

Burada SG kullanımının bize sağladıkları;

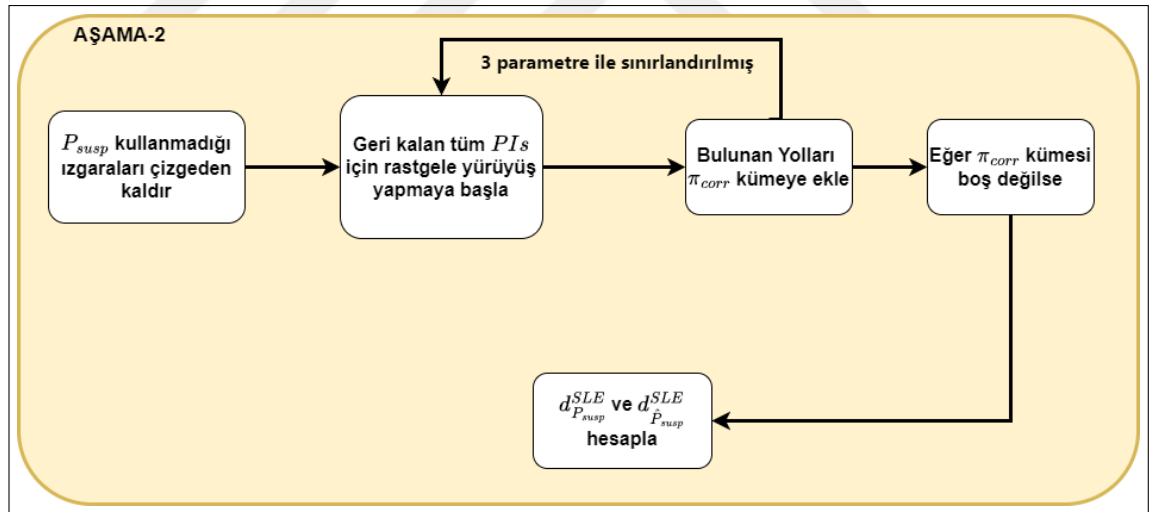
- yol çiftindeki iki yol, ne kadar çok ortak ızgarayı kullanırlarsa, o kadar çok konumsal korelasyona sahip olmaktadır.
- yol çifti konumsal korelasyondan ne kadar benzer şekilde etkilenirse, Truva atının tespit edilebilirliğinin de artması anlamına gelmektedir.

4.3. Rastgele Uyarlanabilir Yöntem ile Truva Atı Tespiti(RUY)

Bu yöntem de dört aşamadan oluşmaktadır. Aşama-0 Bölüm 4.1.1, Aşama-1 Bölüm 4.1.2, önceki yöntemlerle aynıdır. Aşama-2 ve Aşama-3'teki yeni yaklaşımlar sayesinde ciddi oranda hızlanma ve sonuçlarda doğruluk artışı sağlanmıştır.

4.3.1. İlişkili Yol Adayları Kümesi (π_{corr}) Oluşturma

Aşama-1 de şüpheli yol elde edilğine göre, ilişkili yol adaylarını arayabiliriz. Her şüpheli yol için, Aşama-2 de ilişkili yol adayları seçilir. İlişkili yol adaylarının hesaplanması şu şekilde yapılmaktadır: İlk olarak, şüpheli yolun bulunduğu ızgaralar işaretlenir. Daha sonra, devre grafiğinden işaretlenmeyen ızgaralarda bulunan tüm düğümler kaldırılır. Örneğin Şekil 4.4'te, P_{susp} koyu renkli ızgaralarda bulunmaktadır. Devrenin tamamı 64 ızgaraya bölünmüştür (Şekil 3.3). P_{susp} , 64 ızgaradan üçünü kullanıyorsa, devre çizgesinden diğer 61 ızgaradaki tüm düğümler kaldırılmaktadır.



Şekil 4.12: İlişkili Yol Adayları Kümesi Yol Oluşturma.

Tüm süreç Şekil 4.12 de gösterilmektedir. İlk adımda, Birincil girişlerin, kalan kapılardan herhangi biriyle bağlantısı olmayanları da elenmektedir. Ardından, kalan birincil girişler için rastgele yürüyüş algoritmasına [76] başlanmaktadır. Algoritma, üç parametre almaktadır. İlki, birincil girişlerin sayısıdır; ikincisi, maksimum fanout dallarının sayısıdır ve üçüncüsü, bir yol üzerindeki maksimum kapı sayısıdır. Rastgele yürüyüş,

her biri bu üç parametreden biriyle sınırlı olan üç “for” döngüsünden oluşmaktadır. Maksimum fanout dalları, devredeki herhangi bir düğüm için maksimum çıkış bağlantısı sayısıdır. Bir yol üzerindeki maksimum kapı sayısı, yine belirli devre için bir yol üzerindeki maksimum kapı sayısıdır. Sonuç olarak, rastgele yürüyüş her bir birincil giriş için bir kez başlamaktadır, maksimum fanout dallarının sayısı kadar rastgele bir komşu seçmekte ve son olarak bir yol üzerindeki maksimum kapı sayısına ulaşılan kadar rastgele bir başka komşu bulmaya devam etmektedir. Yürüyüş bir birincil çıkışa ulaştığında, o yolu π_{corr} kümesine eklemektedir.

Kullandığımız rastgele yürüyüş algoritması, devre topolojisine ait olan üç parametre olarak çalışma zamanını sınırlamaktadır. Böylece, büyük devreler için çok fazla yol oluşturabilecek özyinelemeli (ing. recursive) yol oluşturma yöntemine göre daha avantajlıdır. Özyinelemeli arama, büyük devreler için oldukça sorunludur ve gereksiz bir çaba içermektedir. Çünkü kalan ızgaraların içerdiği tüm yollara ihtiyacımız yoktur. Sadece Aşama-2 sonunda bir adet ilişkili yola ulaşmak için yeterli sayıda aday elde etmek istiyoruz. Yaptığımız deneyler sonucunda rastgele yürüyüş algoritmasının, hem çok daha iyi performans gösterdiğini, hem de neredeyse tüm şüpheli yollar için yeterli aday sağladığını göstermekteyiz.

Aşama-3’e geçmeden önce, her bir örnek X_i için yol gecikmesi P_{susp} , Aşama-0’da SLE kapı gecikmeleri kullanılarak hesaplanmaktadır. Örnek noktalarının yarısı Truva atsızdır, diğer yarısına Truva atı eklemek için Şekil 3.4’deki yük şüpheli yola eklenir. Bu nedenle, Truva atsız örnekler için $d_{P_{susp}}^{SLE}$ ve Truva atlı örnekler için $d_{\hat{P}_{susp}}^{SLE}$ hesaplanır (denklemler 3.3, 3.4). Hesaplanan gecikme değerleri, kapıların gecikme değerlerinin toplanmasıyla elde edilmektedir. Bu değerler, Aşama-3 tarafından kullanılmak için hazırlanmaktadır.

π_{corr} içinde çıkarılan yolları ilişkili yol adayları olarak adlandırmaktayız. Bunların çoğu şüpheli yol ile yeterli korelasyona sahip olmayabilir ve bu nedenle Truva atı tespiti için kullanılamaz. Bölüm 4.3.2’de, π_{corr} içinden nasıl bir adet ilişkili yol seçildiği açıklanmaktadır.

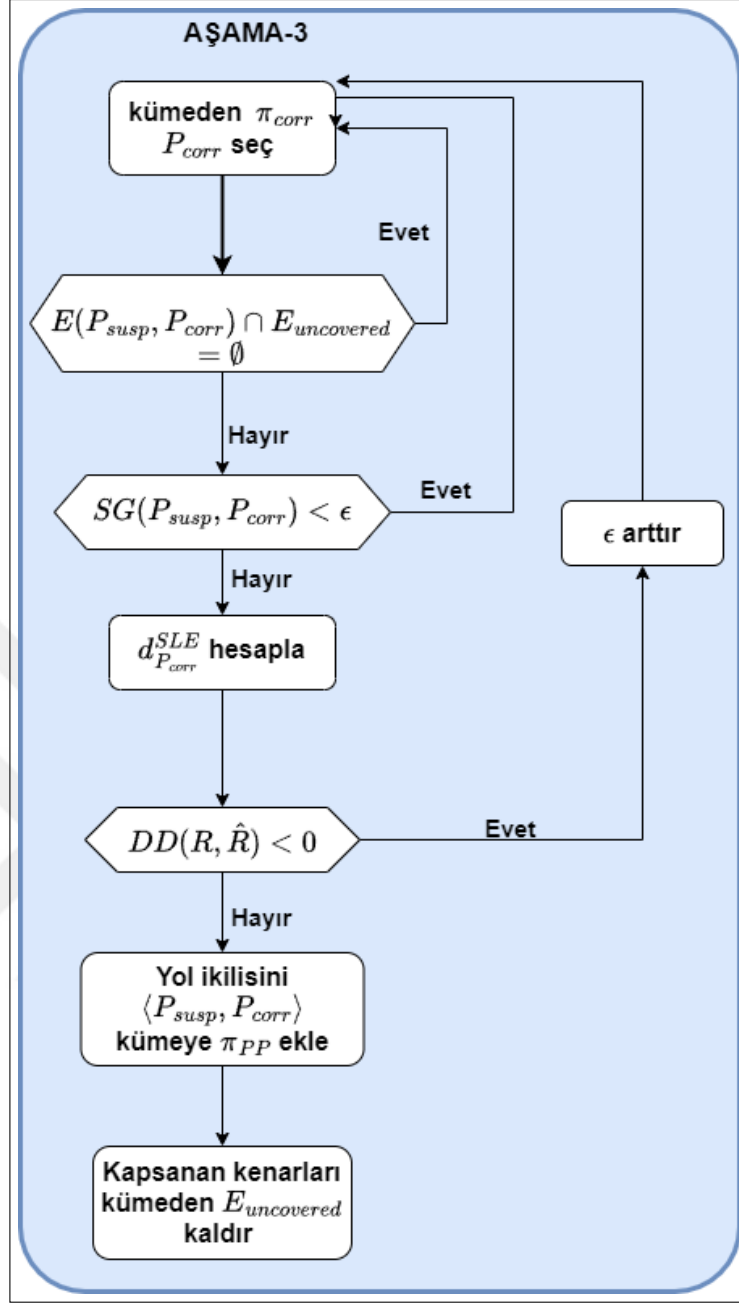
4.3.2. Yol İkili ($\langle P_{susp}, P_{corr} \rangle$) Seçimi

Bu aşama algoritmanın en kritik kısmını oluşturmaktadır. Aynı zamanda, çalışma zamanı açısından darboğaza sokan kısımdır.

Öncelikle, π_{corr} kümesinden bir aday yol, P_{corr} seçilmektedir. Seçilen P_{corr} ile oluşturulan çiftin kenar kapsamımızı artırıp artırmayacağını kontrol etmek zorundayız. Bu amaçla, “henüz” kapsanmamış kenarların kümesi $E_{uncovered}$ ile $\langle P_{susp}, P_{corr} \rangle$ çifti tarafından kapsanabilir kenarların kesişiminin boş bir küme olup olmadığını kontrol etmekteyiz. Eğer bu kesişim boş bir küme ise, bu çiftin kenar kapsamımızı artırmayacağı anlamına gelmektedir ve bu adayı atarak π_{corr} kümesinden başka bir yol seçebiliriz. Bir çift için kapsanabilir kenarlar kümesi, yani $E(P_{susp}, P_{corr})$, her iki yol tarafından da paylaşılmayan kenarların toplamına eşdeğerdir, yani Tanım 3.5’deki gibi paylaşılmayan kenarlardır. Çünkü, bu çiftin gecikme oranını kullanarak, sadece paylaşılmayan kenarların herhangi birine eklenmiş olan Truva atları tespit edilebilmektedir. Aksi takdirde, Truva atının gecikme oranı üzerindeki etkisi fark edilemez.

Eğer Şekil4.13’de bulunan $E(P_{susp}, P_{corr}) \cap E_{uncovered} = \emptyset$ koşulu sağlanmıyorsa, paylaşılan ızgaraların oranını (SG) hesaplamaktayız. SG , her iki yol tarafından paylaşılan ızgaraların sayısının, şüpheli yol tarafından kullanılan ızgaraların sayısına bölünmesi ile elde edilmektedir. Tanım gereği, her zaman 1 değerine eşit veya daha küçük olup, çift içindeki iki yol arasındaki korelasyon miktarı hakkında bir fikir sahibi olmanın kolay bir yoludur [75]. Çünkü, eğer iki yol daha fazla paylaşılan ızgaraya sahipse, bu, varyasyonlardan daha benzer şekilde etkilenecekleri anlamına gelmektedir. Ancak SG , iki yol arasındaki gecikme korelasyonunu anlamak için tek başına kullanılamaz. Çünkü korelasyon, yol içindeki kapı türleri ve kapı sayısı gibi farklı parametrelerden etkilenebilen daha karmaşık bir metriktir. Bu nedenle, SG üzerine ϵ ile temsil edilen bir eşik koymaktayız. Şekil 4.13’de gösterilen ikinci kısımda, adayın ϵ ’dan büyük bir SG ile sonuçlanıp sonuçlanmadığına bakmaktayız. Eğer aday bu koşulu sağlayamıyorsa, büyük olasılıkla şüpheli yol ile iyi bir korelasyona sahip olmadığı için o adayı yine atıyoruz. DD hesaplaması, SG hesaplamasından daha maliyetli olduğu için, ikinci kısımdaki yapı ön eleme olarak kullanılır. İlk SG eşiği oldukça küçük bir sayı olarak alınabilir, çünkü eşik gerektiğinden küçük çıktığında adaptif olarak artırılmaktadır.

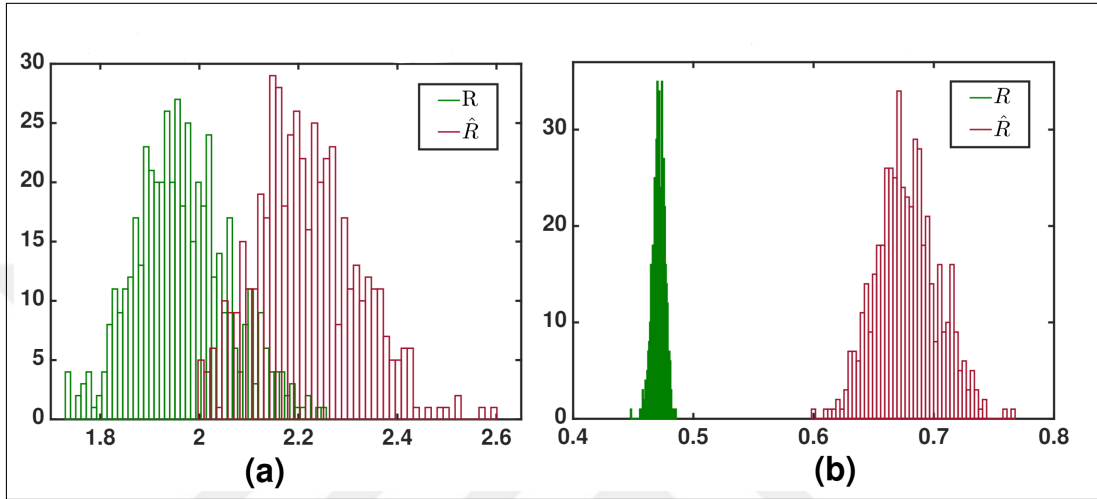
Eğer Şekil4.13’de bulunan $SG(P_{susp}, P_{corr}) < \epsilon$ koşulu sağlanmıyorsa, Aşama-0’da bir



Şekil 4.13: Yol İkili Oluşturma.

kez hesaplanan SLE kapı gecikmeleri, Aşama-2’de P_{susp} için yapıldığı gibi her örnek için P_{corr} gecikmesini ($d_{P_{corr}}^{SLE}$) hesaplamak için kullanılmaktadır. Bu adayın Truva atı tespiti için kullanılıp kullanılmayacağını anlamak için, Tanım 3.5 ile tanımlanan dağılım farkını (DD) hesaplamaktayız. Eğer Truva atı içermeyen (R) ve Truva atı eklenmiş ($\hat{R}$) örnekler için yol gecikme oranlarının dağılım farkı pozitifse, bu çift için Truva atı eklemesinin oran dağılımlarını Truva atını tespit edecek kadar ayırdığı yani çift dağılım modeli olduğu anlamına gelmektedir.

Örneğin, Şekil 4.14’de, c7552 devresinden iki farklı yol çifti için gecikme oranı dağılımlarını görmekteyiz. Soldaki (a) negatif DD durumuna aitken, sağdaki (b) pozitif DD durumuna aittir. Yöntem, Şekil 4.14 (a) ile sonuçlanan ilişkili yol adayını atmakta, ancak Şekil 4.14 (b) ile sonuçlanan adayı seçerek şüpheli yol ile eşlemekte ve ortaya çıkan yol çiftini küme içerisine eklemektedir.



Şekil 4.14: Yol çifti sonuçları; (a): DD negatif , (b): DD pozitif.

Eğer bir ilişkili yol adayı negatif DD ile sonuçlanırsa, bu SG eşiği ϵ 'ın yeterince yüksek olmadığını göstermektedir. Bu nedenle bu eşik artırılmaktadır, ϵ eşiği yalnızca ayrık değerler alabilmektedir. Örneğin, şüpheli yol k adet ızgara kaplıyorsa, ϵ 'ı $1/k$ aralıklarıyla artırmaktayız. Bu, sıfır ϵ ile başlasak bile, $1/k$ hesaplamasından sonra, hiçbir aday uygun değilse, ϵ 1 olur. Şüpheli yol ile aynı ızgaraları paylaşmayan adayları eleme sürecine başlar. Bu uyarlanabilir yaklaşım ile, Aşama-3'ün en maliyetli iki adımı olan gecikme ve dağılım farkı hesaplama adımlarının, adayların çoğu için çalışması önlenmektedir.

Sonunda, bir aday pozitif DD koşulunu sağlarsa, arama durur ve ilgili yol çifti π_{PP} kümesine eklenir. Ardından, bu çift tarafından kapsanabilen tüm kenarlar, yani $E(P_{susp}, P_{corr})$, kapsanmamış kenarlar kümesinden, $E_{uncovered}$, çıkarılır, çünkü artık bu kenarlar için çift bulmamıza gerek yoktur. Bu nedenle, bu çıkarılan kenarlar için, bir şüpheli yol çıkarmamıza veya aday aramamıza gerek yoktur. Yöntem, geri kalan kapsanmamış kenarlar için yol çiftleri bulmak üzere Aşama-1'in ilk adımına geri döner.

Aksi takdirde, tüm adaylar tükendiği halde hiçbir aday uygun değilse, ilgili şüpheli

kenarı işaretleyerek Aşama-1'in ilk adımına geri döneriz. Böylece o kenar için tekrar yol çifti bulmaya çalışmayız. Ancak bu, o işaretli kenarı kapsayamayacağımız anlamına gelmez. Çünkü başka bir şüpheli kenar için tespit edilen herhangi bir çift, aynı zamanda o işaretli kenara eklenen Truva atını tespit etmek için de uygun olabilir. Yani işaretli kenar başka bir şüpheli kenarın $E(P_{susp}, P_{corr})$ kümesinin bir elemanı olabilir. Üretim öncesi döngüler, tüm şüpheli kenarlar çıkarılana veya işaretlenene kadar devam eder.

Aşama-3'ün sonunda, bir dizi ilişkili yol çifti oluşturulmaktadır. Ancak, bu kümede bazı gereksiz yol çiftleri bulunabilir. Örneğin; N adet yol çiftinin içerdikleri kenarların birleşimi şüpheli kenarların tamamını kapsıyor olsun. Bu durumda, başka bir yol çifti seçilse bile yeni kenarları kapsayamayacaktır. Bunun sonucunda, N adetten sonra eklenen yol çiftleri fayda sağlamamış olurlar. Bu yol çiftlerinin gecikmelerini ölçmek için de, üretim sonrası testler sırasında ayrıca ek bir çaba gerekmektedir. Hem her çift için kenar kapsamı hem de evrensel kenar kümesi bilindiğinden, örnekteki gibi fayda sağlamayan yol çiftlerinden kurtulmak için π_{PP} üzerinde iyi bilinen minimum küme kapsama (ing. min set cover) açgözlü algoritmasının doğrudan bir uygulaması mümkündür.

Bu küme, yerleşim planı hazır olur olmaz oluşturulabilir. Truva tespiti yongaların üretiminden sonra, yol çiftleri kümesindeki yolların gecikmelerini hesaplayarak yapılmalıdır. Üretim sonrası aşama-4, Bölüm 4.3.4 açıklanmaktadır.

4.3.3. Üretim Sonrası Testlerin Hızlandırılması

Yöntemlerimizin hepsinde aynı hedefe sahip üç adım bulunmaktadır. Üçüncü adım sonlandığında, elimizde test yapılması gereken yol çiftleri olmaktadır. Üretim sonrası testler (ing. post-manufacturing tests) sırasında daha az sayıda yol çifti ile aynı sayıda devre kenarını kontrol edebilir miyiz?" sorusundan yola çıkarak, minimum küme kapsama (ing. set-cover) yöntemini kullanmaya başladık [77]. Bu yöntem, Aşama-3'ten sonra kullanılmaktadır. Ana amaç, üretim sonrası testlerden önce minimum sayıda yol çifti elde etmektir.

Yol çiftlerinin sayısı, üretim sonrası aşama için oldukça önemli olmaktadır. Ne kadar az yol çiftimiz olursa, üretim sonrasındaki test süreci o kadar kısa sürmektedir. Amacımız, seçtiğimiz çiftlerin, girdi çiftlerinde bulunan tüm kenarları içermesi için minimum

Tablo 4.1: Minimum Küme Kapsama Yönteminin Çalışmasına Dair Örnek.

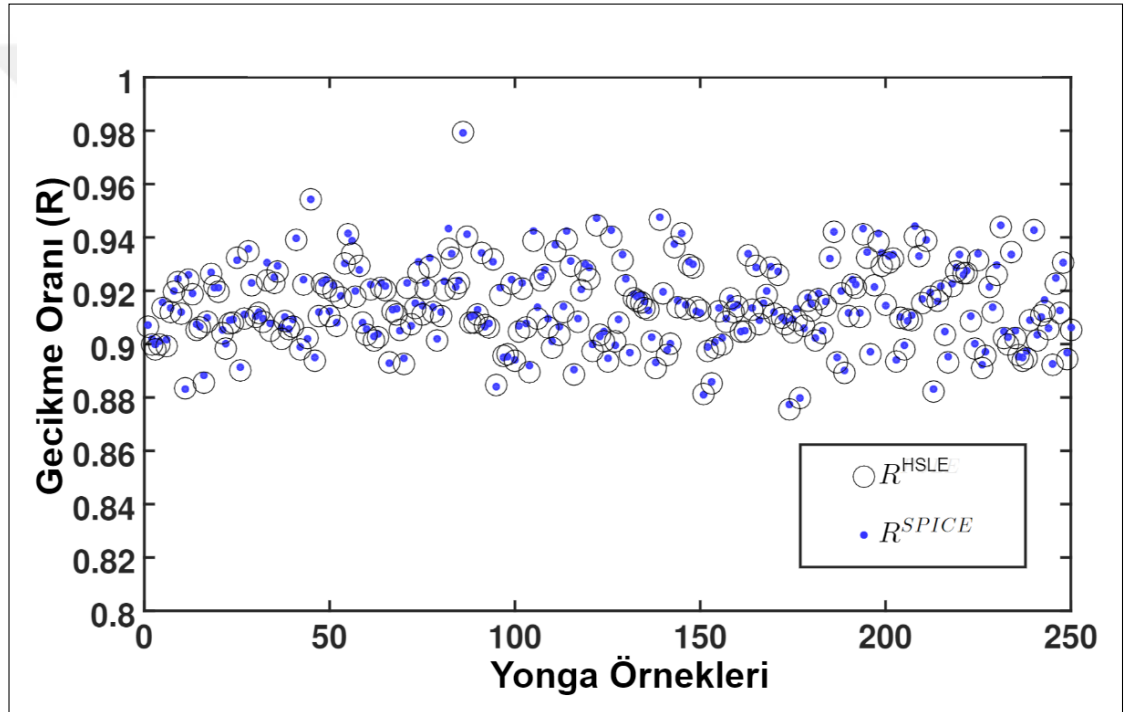
P_{susp}	P_{corr}	Paylaşılmayan Kenar Sayısı Adım1	Paylaşılmayan Kenar Sayısı Adım2	Paylaşılmayan Kenar Sayısı Adım3	Paylaşılmayan Kenar Sayısı Adım4
P1	P2	5	0	0	0
P1	P5	4	2	0	0
P2	P4	4	2	1	0
P3	P4	2	2	2	0
P2	P3	4	2	1	0

sayıda yol çifti seçmektir. Elde edilen sonuçlar üzerinde bir kapsam kümesi algoritması kullanıldığında, kapsadığı kenar sayısı sabit kalarak yol çifti kümesinin boyutunun azaldığını fark etmiş bulunmaktayız.

Açgözlü (ing. greedy) minimum küme kapsama algoritması [77], kenarların genel kümesini tamamlayana kadar aramaya devam etmektedir. Tablo 4.1’de her satır, şüpheli bir yol ve ilişkili yol adayını içeren yol çiftini temsil etmektedir. Bu örnekteki yollar Şekil 3.1’den alınmıştır. Yollar ve içerdikleri kapılar şu şekildedir. $P1 : < n1, n2, n3, n4 >$ $P2 : < n11, n10, n4 >$ $P3 : < n5, n6, n7 >$ $P4 : < n9, n6, n7 >$ $P5 : < n1, n2, n6, n7 >$ Tabloda üçüncü sütundan itibaren gösterilen her yinelemede, algoritma tüm yol çiftlerini kontrol etmekte ve en fazla paylaşılmayan kenara sahip olanı seçmektedir. Bu seçim tabloda renklendirilerek gösterilmiştir. Her bir seçimin ardından toplam küme ve veri kümesindeki paylaşılmayan kenarların sayısını güncellemektedir. Bu işlem kenarların tamamı kapsanıncaya kadar sürdürülmektedir.

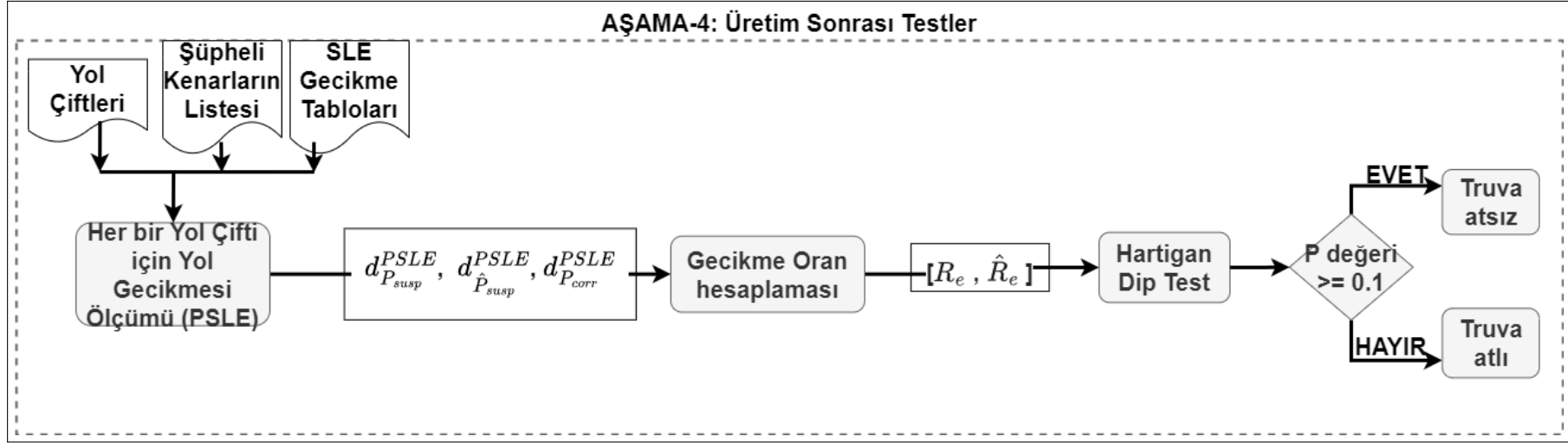
4.3.4. Üretim Sonrası Truva Atı Tespiti

Yöntemimizin önerdiği yolun doğruluğunu, şüpheli bir yol ve şüpheli yola bağlı olan bir yol çiftinin gecikmelerini hem *HSLE* modeli hem de SPICE simülasyonları ile hesaplayarak doğruladık. Şekil 4.15’de gösterildiği gibi, 250 örnek kullanılmıştır. Şüpheli yol ve bağlantılı yolun gecikmeleri, her örnek için hem *HSLE* hem de Spice ile hesaplanmıştır. Şekildeki gecikme oranlarının neredeyse mükemmel şekilde eşleştiğini görmekteyiz. Böylece bu aşamada kullandığımız *HSLE* modelinin SPICE kadar doğru sonuçlar verdiğini doğrulamış olduk.

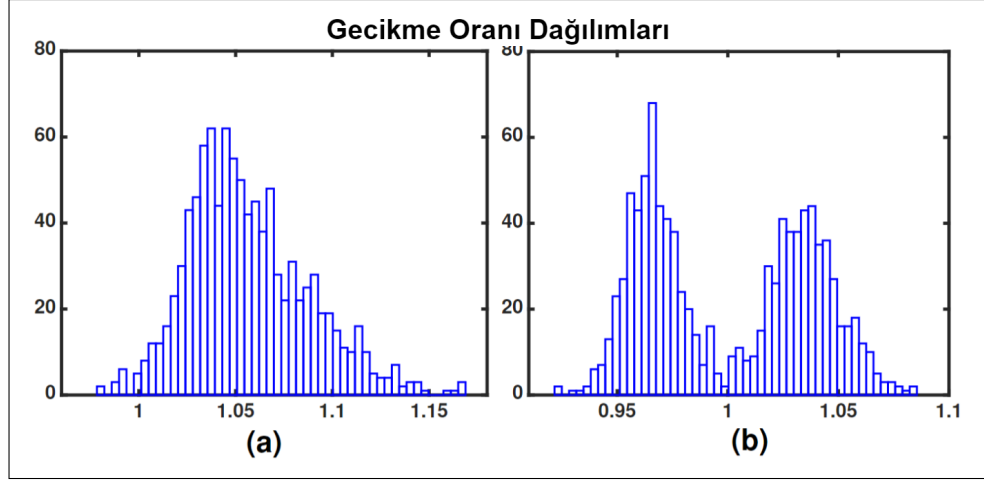


Şekil 4.15: HSLE modelinin SPICE ile doğrulanması.

Şekil 4.16’da gösterilen üretimden sonrası Truva atı tespiti, yol çiftlerindeki yolların gecikmelerinin hesaplanmasıyla gerçekleştirilmelidir. Bu çalışmada, yongaları gerçekten üretmediğimiz için hassas olan *HSLE* gecikme oranı hesaplamasını kullanmaktayız. Ancak, gecikme oranlarının nasıl hesaplandığı veya ölçüldüğünden bağımsız olarak, gecikme oran dağılımlarının tek modele mi çift modele mi ait olduğunu, test eden taraftan kontrol edilmektedir. Bu amaçla Hartigan’ın çift model [78] testini kullanmaktayız. Bu teste göre, bir p-değeri hesaplanır ve bu değer 0.1’den küçükse, bu bimodal bir dağılımı işaret etmektedir.



Şekil 4.16: Üretim Sonrasında Truva Atı Tespiti.



Şekil 4.17: c3540 devresinden çıkarılmış yol ikilileri: (a): Truva atı tespit edilemedi, (b): Truva atı tespit edildi.

Örneğin, Şekil 4.17, c3540’ın iki farklı yol çiftinden elde edilen iki gecikme oranı dağılımını göstermektedir. Her çift için 1000 örnek varsayılmakta ve bunların yarısında çiftin “paylaşılmayan kenarın” bir Truva atı yerleştirilmiştir. Şekil 4.17 (a)’da p-değeri 0.43 olup unimodal olarak işaretlenmiştir, oysa Şekil 4.17 (b)’de p-değeri 0 olup bimodal olarak işaretlenmiştir. Diğer bir deyişle, Şekil 4.17 (a)’daki çift için yöntem başarısız olmuş ve Truva atı tespit edilememiştir, oysa Şekil 4.17 (b)’deki çift için önerilen yöntem Truva atını başarıyla tespit etmiştir.

Yol çifti gecikme oranının Truva atı varlığını ortaya çıkarması için, Truva atının yol çiftin paylaşımsız kenarlarından birine yerleştirilmiş olması gerekmektedir. Böylece yol çiftinin gecikme oranı, Truva atı etkisini ortaya çıkarabilir. Bu çalışmada önerilen yöntem ile kritik kenarların kümesi ortaya çıkarılır ve yalnızca bu kenarlara yoğunlaşarak Truva atının tam yeri belirlenebilir.

5. Deneysel Sonuçlar

Bu çalışmadaki KKY ve PIY deneyleri için, gerekli hesaplamalar ve simülasyonlar Xeon E5-2620, altı çekirdekli, 2 GHz işlemcili ve 24 GB RAM'li HP z620 iş istasyonunda gerçekleştirilmiştir. Önerilen yöntemin sınırlarını test etmek ve tespit performansını görmek için bir XOR kapısından oluşan çok küçük boyutlu bir Truva atı 3.4 kullanılmıştır. Gecikme ölçümleri için, Bölüm 3.2deki varyasyon modeli uygulanmış, 45 nm standart hücre kitaplığına göre sentezleme bilgileri ve Bölüm 3.3deki *SLE* yöntemi kullanılmıştır. Belirli bir test devresi için kapıların konumsal bilgilerini elde etmek için DEF dosyaları [79] kullanılmıştır.

5.1. Korelasyon Katsayısına Dayalı Truva Tespit Yöntemi Sonuçları

Tablo 5.1 KKY sonuçlarını göstermektedir. Devredeki toplam kenar sayısı, elde edilen kapsanabilir kenar sayısı ve toplam ilişkili yol adaylarının sayısı Tablo 5.1'in ilgili sütunlarında yer almaktadır. Elde edilen yol çiftlerinin sayısı, şüpheli kenarların sayısına eşittir, bu nedenle tabloda belirtilmemiştir.

Devrenin tamamen kapsanabilir olması, herhangi bir kenara yerleştirilen herhangi bir Truva atının yöntem tarafından tespit edilebileceği anlamına gelmektedir. Sonuçlar, tabloda yer alan karşılaştırma devreleri incelendiğinde kapsanabilirlik oranının ortalama 90% civarında olduğunu göstermektedir. Aynı zamanda KKY için gereken korelasyonlu yol adaylarının sayısını da göstermektedir. Bu da Bölüm 4.1'de bahsedilen KKY yöntemine ait Aşama-3 kısmını en maliyetli kısım haline getirmektedir. Çünkü her bir ilişkili yol adayı için yol gecikmesi ve korelasyon katsayısı Aşama-3 de hesaplanmaktadır.

Tablo 5.2, Bölüm 4.1'da açıklanan KKY'nin her aşaması için - imalat sonrası kısım olan Aşama-4 hariç - devrelerin sahip olduğu tüm kenarlar üzerine uygulandığında harcanan zaman miktarını göstermektedir.

KKY yönteminin başlıca dezavantajı Aşama-3 hesaplamalarından kaynaklanan hesaplama karmaşıklığıdır (ing. computational complexity). Çünkü bu aşama, her bir örnek yonga ve çıkarılan ilişkili yol adaylarının her biri için (Denklem 3.2) kullanarak yol gecikmesini

Tablo 5.1: KKY Yöntemine ait Sonuçlar.

Devreler	Özelliği	Kapsanan Kenar Sayısı	Kenar Kapsamı %	Aday Sayısı (KKY)
432	27 kanallı kontrolcü	255	95.7%	36,647
499	32bit SEC	296	64.5%	29,320
880	8bit ALU	507	94.7%	39,176
1355	32bit SEC	856	96.6%	17,200,357
1908	16bit SEC	1420	93.4%	5,353,429
2670	12bit ALU	1850	95.2%	7,619,334

Tablo 5.2: KKY Yönteminin Zaman Kullanımı.

Devreler	Çalışma Zamanı Aşama-1	Çalışma Zamanı Aşama-2	Çalışma Zamanı Aşama-3
432	4sn	54sn	110sn
499	7sn	38sn	102sn
880	23sn	180sn	169sn
1355	1.2dk	13.21saat	21.76saat
1908	1.38dk	4.96saat	7.63saat
2670	7.2dk	6.39saat	9.18saat

hesaplamaktadır. Daha sonra, bu yol gecikmeleri kullanılarak her aday için korelasyon katsayısı tekrar hesaplanmaktadır. Ancak, Tablo 5.1’de gösterildiği gibi korelasyonlu yol adaylarının sayısı, artan devre boyutu veya karmaşıklığı nedeniyle oldukça büyük olabilmektedir. Ayrıca, doğru sonuçlar elde etmek için örnek sayısının yeterince büyük olması gerekmektedir, bu da Aşama-3 hesaplamalarını karmaşık hale getirmektedir.

Aşama-3 için ortaya çıkan hesaplama karmaşıklığını inceleyelim. Bu denklemde, ÖS çip sayısını, $\text{KS}_{\text{tüm}}$ tüm devredeki lojik kapı sayısını, $\text{Maliyet}_{\text{SLE}}$ SLE hesaplama maliyetini (Denklem 3.2) (iki çarpma ve bir toplama işleme), AS ilişkili yol adaylarının sayısı, KS_{ort} tüm ilişkili yol adayları üzerindeki ortalama lojik kapı sayısı, $\text{Maliyet}_{\text{toplama}}$ yol gecikmesi hesaplamasında kullanılan bir toplama işleminin maliyetidir. $\text{Maliyet}_{\text{kk}}$ korelasyon katsayısı hesaplaması için birim maliyettir. Bunun için, toplama, bölme ve karekök gibi aritmetik işlemleri kullanır. Son olarak $\text{Maliyet}_{\text{karşılaştırma}}$ aralarındaki en büyük değeri bulmak için kesirli sayıları (ing. floating point) karşılaştırmanın maliyetidir.

$$\begin{aligned} \text{Maliyet}_{\text{aşama}(iii)}^{\text{KKY}} = & \text{ÖS} \times \text{KS}_{\text{tüm}} \times \text{Maliyet}_{\text{SLE}} + \\ & \text{AS} \times \text{ÖS} \times \text{KS}_{\text{ort}} \times \text{Maliyet}_{\text{toplama}} + \\ & \text{AS} \times \text{ÖS} \times \log(\text{ÖS}) \times \text{Maliyet}_{\text{kk}} + \\ & \text{AS} \times \text{Maliyet}_{\text{karşılaştırma}} \end{aligned} \quad (5.1)$$

Denklemin ilk satırı her bir kapı ve her bir örnek yonga için SLE gecikmelerini hesaplama maliyetini, ikinci satır ise ilk satırda hesaplanan SLE gecikmelerini kullanan ve her bir yonga ve her bir korelasyonlu yol adayı için gerçekleştirilen yol gecikmesi hesaplamalarının maliyetini gösterir. Denklemdeki üçüncü satır, ikinci satırda hesaplanan yol gecikmelerini kullanarak korelasyon katsayısı hesaplamasını ifade etmektedir. İkinci satır ve özellikle üçüncü satır karmaşıklığın ana kaynağını oluşturmaktadır. Son satır ise maksimum korelasyon katsayısına sahip adayı bulmak için kullanılmıştır.

5.2. Paylaşımlı Izgaraların Oranına göre Truva Atı Tespit Yöntemi Sonuçları

SG değerinin hesabı için her iki yolun da bulunduğu toplam ızgara sayısının ($\text{Maliyet}_{\text{birleşim}}$) ve her iki yol tarafından ortak kullanılan ızgaraların ($\text{Maliyet}_{\text{kesişim}}$) tespit edilmesini

gerektirir. Ardından, SG değerini elde etmek için tek bir bölme işlemi ($\text{Maliyet}_{\text{bölme}}$) yeterlidir. Sonuç olarak, SG değerini elde etmek için hiçbir yol gecikmesi ve maliyetli korelasyon katsayısı hesaplaması gerekmemektedir. Ayrıca, SG değeri yongadan yongaya değişmediği için her örnek için yeniden hesaplama gerekmez. Sonuç olarak, Aşama-3'ün yeni maliyeti (5.2) ile gösterildiği gibi yazılabilmektedir. IS_{avg} bir yol çifti tarafından kullanılan ortalama ızgara sayısını temsil etmektedir. Kesinlikle IS_{ort} örnek sayısından çok daha küçüktür. $\text{AS} \times \text{Maliyet}_{\text{karşılaştırma}}$, (5.1)'ye benzer şekilde maksimum SG^i ile ilişkili yol adayı i 'yi bulmak için kullanılmaktadır.

$$\text{Maliyet}_{\text{aşama(III)}}^{\text{PIY}} = \text{AS} \times \text{IS}_{\text{ort}} \times (\text{Maliyet}_{\text{kesişim+birleşim+bölme}}) \quad (5.2)$$

$$+ \text{AS} \times \text{Maliyet}_{\text{karşılaştırma}}$$

Özellikle SG denkleminin paydası küçük bir sayı olduğunda, birden fazla ilişkili yol adayı en büyük SG değerine sahip olabilir. Böyle bir durumda, en iyi aday, yalnızca en büyük ortak ızgara oranına sahip adaylar arasında yol gecikmeleri hesaplanarak tespit edilebilir. Bu maliyetin (5.2)'a eklenmesi gerekmektedir. Ancak bunu teorik olarak göstermek zordur çünkü aynı en büyük ortak ızgara oranına sahip olan, bu tür adayların sayısı önceden bilinmemektedir. Ayrıca, Tablo 5.3 bu nitelikteki aday yolların toplam sayısını son sütunda bildirmektedir.

Tablo 5.3, Bölüm 4.2'da önerilen PIY yöntem yöntemine ait karşılaştırma sonuçlarını göstermektedir. Tablodan yapılan ilk çıkarım, hem KKY hem de PIY'in neredeyse tüm devredeki kenarları kapsayabildiği veya güvence altına alabildiği ve bunun da yaklaşık %90 kenar kapsaması ile sonuçlandığıdır. Bu da, her iki yönteminde devrenin %90'ında herhangi bir yere yerleştirilen herhangi bir Trojan'ı tespit edebileceği anlamına gelir. Bariz bir aykırı devre olan c499 hariç tutulduğunda, PIY'in kenar kapsama alanı ortalama olarak yaklaşık %93 olmaktadır.

Ocak 2008'de, DARPA'nın Mikrosistemler Teknoloji Ofisi müdür yardımcısı Collins, üç şirket (Raytheon, Luna Innovations ve Xradia) arasında bir Truva atı tespit yarışması başlatır. Truva atı devresi MIT Lincoln Labs tarafından yerleştirilir. Collins, IEEE Spectrum dergisine hedefin yüzde 90 tespit oranı olduğunu belirtir [80], bu da yüzde 90 kapsama alanının yeterliliğini teyit etmektedir.

Son iki sütun sırasıyla KKY ve PIY için incelenmesi gereken ilişkili yol adaylarının sayısını göstermektedir. PIY için aday sayısı 100 kattan daha azdır çünkü her bir şüpheli

yol için en büyük ortak ızgara (SG) oranına sahip olanlar dışındaki tüm adaylar elenmektedir. Bu tablo, PIY'in çok daha küçük bir ilişkili yol adayı kümesi için yol gecikmesi hesaplamaları yapmasına rağmen, seçilen yol çiftlerinin doğruluğunu kaybetmediğini göstermektedir.

Tablo 5.3: KKY ve PIY Sonuçlarının Karşılaştırılması.

Devreler	Kenar Kapsamı	Kenar Kapsamı	Aday Sayısı	Aday Sayısı
	KKY	PIY	KKY	PIY
432	95.7%	92.5%	36,647	580
499	64.5%	56.7%	29,320	521
880	94.7%	92.5%	39,176	1782
1355	96.6%	95.8%	17,200,357	102,781
1908	93.4%	91.5%	5,353,429	19,583
2670	95.2%	93.5%	7,619,334	77,390

Tablo 5.2'den de anlaşılacağı üzere, KKY ile konumsal korelasyon tabanlı Truva atı tespitinin en çok zaman alan kısmı Aşama-3 dür. Bu nedenle PIY bu aşamayı hızlandırmak için önerilmiştir. Aşama (iii) hesaplamalarında PIY'in KKY'ye göre hızlanmasını ölçmek için, her iki yöntemle Aşama-3 hesaplaması için gereken süreyi kaydettik. Şekil 5.1, her bir kıyaslama devresi için elde edilen Aşama-3 hızlanmasını bir çubuk grafik olarak göstermektedir. Test devreleri üzerinde yapılan ölçümlere göre, Aşama-3 PIY yöntemi sayesinde, KKY'nin ortalama olarak yaklaşık 9 kat hızlandırılmıştır. Bu ciddi bir hız artışıdır. Tablo 5.3'ün son sütununda gösterilen adaylardan kaynaklanan tüm ek yol gecikmesi hesaplamalarının Şekil 5.1 ve 5.2'daki hızlanma miktarlarında dikkate alınmıştır.

PIY'nin KKY'ye göre Aşama-1'den Aşama-3'e kadar olan toplam süresi dikkate alınarak elde edilen hız artışı Şekil 5.2'da gösterilmektedir. Kıyaslama devrelerindeki

tüm kenarlar üzerinde çalıştırıldığında, PIY yöntemi ortalama olarak KKY'nin hızını iki katına çıkarabilmektedir. Daha doğru bir ifadeyle, PIY kenar kapsamını sadece %3 azaltarak yaklaşık %100 hız artışı sağlamaktadır. Bu da önerilen yöntemin verimliliğini ve doğruluğunu kanıtlamaktadır.

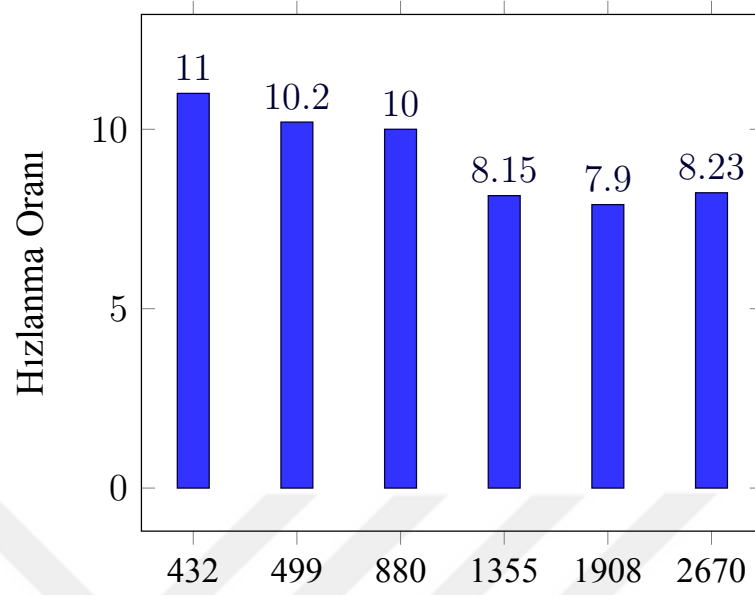
5.3. Rastgele Uyarlanabilir Yöntem ile Truva Atı Tespit Sonuçları

Bu yöntem, devre düzeni bilgilerini kullanarak Truva atı tespitinde kullanılacak bir dizi yol çiftini elde etmeyi amaçlamaktadır. Algoritmanın zaman karmaşıklığı, en karmaşık görevlerin bulunduğu 3. Aşama tarafından tanımlanmaktadır. Yöntemin karmaşıklığı aşağıdaki denklemle ifade edilmektedir:

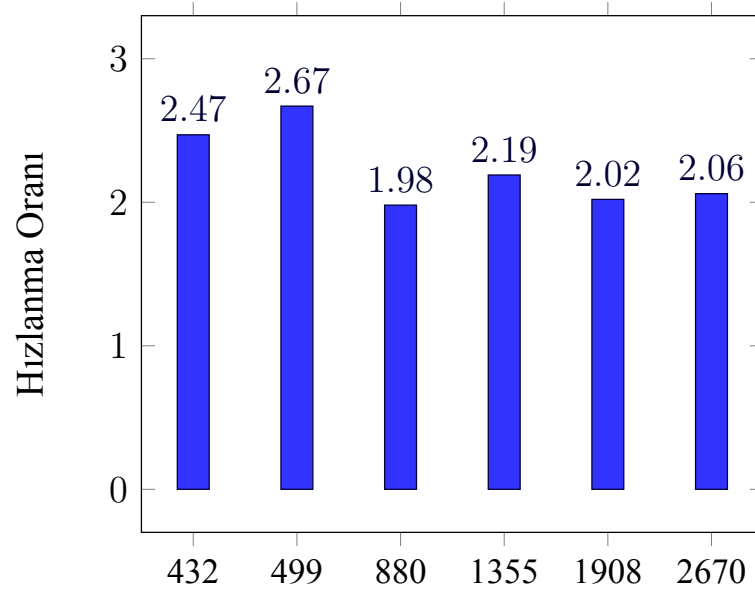
$$O(E * makYolUzunlugu^2 * (ÖrnekSayısı + GirdiSayısı * makFanout)) \quad (5.3)$$

$E(5836)$ dervedeki kenar sayısı, $makFanout(5)$ yolun sahip olduğu maksimum dallanmaya sahip kenar sayısı, $makYolUzunlugu(47)$ kritik yolun sahip olduğu kapı sayısı yani bir yolun sahip olabileceği maksimum kapı sayısı, $ÖrnekSayısı(1000)$ örnek sayısı, $GirdiSayısı(233)$ birincil giriş sayısıdır. Test devreleri için en kötü durum değerleri parantez içinde verilmiştir. Gerçek hesaplama süreleri Tablo 5.5'de bulunmaktadır.

Yöntemin tüm devrelerdeki tespit performansı Tablo 5.4 ile gösterilmektedir. Sütunlar, soldan sağa doğru, devredeki toplam kenar sayısını (şüpheli kenarlar), üretim öncesi aşamada toplanan yol çiftleri ile kapsanabilir kenar sayısını, üretim sonrası simülasyonlarda aynı yol çiftleri ile ortaya çıkan gerçek kapsanan kenarları, set cover algoritması çalıştırılmadan önce ve sonra toplanan yol çiftlerinin sayısını ve son olarak transistör sayısı açısından Trojan'ın göreceli boyutunu temsil etmektedir. Tabloda ÜÖ kısaltması üretim öncesi, ÜS kısaltması üretim sonrasını ifade etmektedir.



Şekil 5.1: Aşama-3 hızlanması: PIY yönteminin KKY'ye göre.



Şekil 5.2: Tüm Aşamaların Hızlanması: PIY yönteminin KKY'ye göre.

Tablo 5.4: Rastgele Uyarlanabilir Yöntemine ait Sonuçlar.

Test Devreleri	Kenar Sayısı	Kapsanan Kenarlar (ÜÖ)	Kapsanan Kenarlar (ÜS)	Kapsanan Kenarlar (ÜS) (%)	Yol Çiftleri	Yol Çiftleri Min. kapsama Sonrası	Truva Atı Boyutu (%)
c880	507	464	462	91.13	172	150	0.555
c1355	856	744	742	86.68	229	150	0.430
c1908	1419	1295	1286	90.63	409	319	0.274
c2670	1850	1719	1687	91.19	565	437	0.176
c3540	2630	2479	2479	94.26	768	596	0.132
c5315	3878	3782	3701	95.45	1439	1192	0.088
c6288	4288	3645	3596	83.86	850	550	0.099
c7552	5836	5668	5663	97.03	1678	1302	0.064

Tablo 5.4, her test devresi için elde edilen kenar kapsama oranlarını bildirerek tam devre üzerinde Truva atı tespit yeteneğini göstermektedir. İlk üç devreyi, yani c17, c432 ve c499, pratik olmayan küçük boyutları nedeniyle hariç tutmaktayız. Sonuçlar, ortalama kapsama veya tespit oranının %91.38 olduğunu göstermektedir.

Tabloyu bir örnekle açıklayalım: c5315 devresi için, birincil giriş ve çıkış kenarları dikkate alınmadan toplam 3878 kenar bulunmaktadır. Üretim öncesindeki aşamaların sonunda elde edilen kapsanabilir kenar sayısı 3782'dir, ancak üretim sonrası simülasyonda, 81 kenar için iki modlu testler ile HT tespit edilememiştir. Bu nedenle, gerçek kapsama 3701 kenar olup bu da %95.45'e karşılık gelmektedir. Bu yüzde, eşdeğer olarak tespit olasılığı olarak da adlandırılabilir. Üretim öncesi aşamalar 1439 yol çifti (2878 yol) ile sonuçlanmakta, ancak açgözlü set cover algoritması uygulandıktan sonra aynı kenar kapsamı için yalnızca 1192 yol çiftinin yeterli olduğu bulunmuştur. Bu nedenle, üretim sonrası test cihazı, devrenin %95.45'ini güvence altına almak için sadece 2384 yol gecikmesini ölçmelidir. Açgözlü set cover algoritmasından sonra yol çiftlerinin sayısındaki azalma yaklaşık %18'dir. Son sütun, Trojan'ın göreceli boyutunu göstermekte olup, c5315'in 11,308 transistörü varken, Trojan'ın sadece 10 transistöre sahip olması nedeniyle %0.1'den bile küçüktür.

Tablo 5.4'de, c5315 için üretim sonrası kapsanan kenarların %95.45 olması, %4.55 kapsanamadığını, yani bu kenarlara Truva atı eklensede tespit edilemeyeceğini ifade etmektedir. Bu nedenle c5315 devresinin, yanlış negatif oranı %4.55'dir. Çift model testine dayalı Truva atı tespitinin (Bölüm 4.3.4) herhangi bir yanlış pozitif sonuç verip vermediğini ortaya çıkarmalıyız. Bunun için, tüm devrelere hiçbir Truva atı yerleştirmeden deneyleri tekrarlarız. Daha sonra, herhangi bir gecikme oranı dağılımı için yanıltıcı bir çift model sonucu olup olmadığını inceleriz. Tüm test devrelerinden elde edilen sonuçlar, yöntemin Truva atı olmadığında hiçbir tespit yapmadığını göstermektedir. Başka bir deyişle, yanlış pozitif oranı %0'dır.

Her aşamanın çalışma süresi Tablo 5.5'de gösterilmiştir. Aşama-0 (Bölüm 4.1.1) hızlı ve yalnızca bir kez çalıştırılan işlemlerden oluşmaktadır. Bu nedenle tabloda rapor edilmemiştir. Tablo, gerekli yol çiftlerinin tüm devreler için bir saatten kısa sürede çıkarıldığını göstermektedir. Daha ilginç olan kısım, çalışma süresinin artış oranının devredeki kapı sayısı ile neredeyse doğrusal oranda olmasıdır.

En fazla zaman alan kısım, ilişkili yol adaylarının ve şüpheli yol gecikme hesaplamalarının yapıldığı Aşama-2'dir. Aşama-3 aslında her aday üzerinde çalışır, ancak uyarlanabilir algoritmanın kullanımı sayesinde oldukça hızlıdır. ϵ eşik değerini kullanan uyarlanabilir algoritma (Bölüm 4.3.2) kullanılmamış olsaydı, Aşama-3'ün ortalama çalışma süresi yaklaşık 8 kat artardı (Şekil 5.1).

Tablo 5.5: Aşamaların Çalışma Zamanları (saniye cinsinden).

Devreler	Kapı Sayısı	Aşama-1	Aşama-2	Aşama-3	Toplam (sn)
880	383	9	8	3	20
1355	546	21	24	14	59
1908	880	21	56	263	100
2670	1193	79	109	38	226
3540	1669	114	284	104	502
5315	2307	448	715	167	1333
6288	2416	280	850	2165	3296
7552	3512	554	1428	417	2399

Bölüm 4.2'deki özyinelemeli yaklaşım ile rastgele yürüyüşü karşılaştırdığımızda büyük bir fark görülmektedir. Eğer özyinelemeli yaklaşım tercih edilseydi, Aşama-2'de toplanan toplam yol adaylarının sayısı 43.030'dan 829.558'e yükselecekti. 43.030, aslında önerilen yöntemin üst sınırıdır, çünkü bu maksimum rastgele yürüyüş arama algoritmasının çeşitli parametrelerle sınırlandırılmasına bağlıdır. Bu parametreler ve detayları Bölüm 4.3.1'de anlatılmaktadır. Yaklaşık 20 kat azalan aday sayısı, hem Aşama-2 hem de Aşama-3 zamanlamalarını iyileştirmiştir.

Rastgele yürüyüş arama algoritması, devredeki maksimum fanout dalları ile sınırlıdır ve c6288 dışındaki devreler için bu değerlerin 8 ile 16 arasında olduğu bulunmuştur. Bu sınırın kapsam ve hız üzerindeki etkisini görmek için farklı deneyler yaptık. c6288

dışındaki tüm devreler için bu sınırı 5 olarak seçersek, üretim öncesi aşamalar tamamlandıktan sonra, elde edilen ortalama kapsanabilir kenar yüzdesi %95,2'den %94,9'a düşerken, Aşama-2'nin ortalama çalışma süresi 392 saniyeden 305 saniyeye düşmektedir. Ayrıca, yineleme sayısını 20'ye çıkardığımızda, kapsanabilir kenar yüzdesi değişmezken Aşama-2'nin ortalama çalışma süresi 420 saniyeye çıkmaktadır. Ancak, c6288 için maksimum fanout dalı sadece üçtür ve bu sayıyı 20'ye çıkardığımızda, kapsanabilir kenar yüzdesi %85'ten %88,9'a çıkarken Aşama-2'nin çalışma süresi 850 saniyeden 1508 saniyeye çıkmaktadır.

5.4. Truva Atının Tetikleyici Kısımının Tespit Edilebilirliği

Buraya kadar elde edilen sonuçların tamamı, yalnızca gecikme oranına eklenen yük kısmının etkisini tespit etmektedir. Ancak, Truva atının bir başka zayıflığı da tetikleme kısmıdır. Truva atı, fonksiyonel testlerde tespit edilmemek için pasif kalmak zorundadır. Truva atımızın tetikleyici kısmı olan VE kapısının giriş sayısı arttıkça, Truva atının etkinleştirilme olasılığı azalır. Bu da mantık bazlı tespiti zorlaştırır. Bölüm 3.4'de anlatılan Truva atı modelimizden yola çıkarak, tetikleyici olarak dört girişli bir VE kapısı, ve devrede tetikleyici VE kapısının girişlerine bağlı dört farklı kenar için ekstra bir yük bulunmaktadır.

Aslında, yük tespiti için zaten ilişkili yol çiftlerini oluşturduk. Bu nedenle, aynı çiftleri kullanarak yol gecikme oranlarındaki farkı tespit edebiliriz. Yine, yük deneyi gibi, her devrenin her kenarını test ederiz. Bu sefer sadece tetikleyici kapıya ek bir bağlantı kenarı sağlayarak, dört bacağında devreye bağlanmasını sağlarız. Ardından, yöntemin yol gecikme oranı farkını tespit edip edemeyeceğini doğrularız. Tetikleyici bağlantısını tespit edebildiğimiz her kenar, kapsanmış olarak işaretlenir. Tablo 5.6, tetikleyici tespiti için elde edilen kenar kapsama oranını ve birleşik tespit olasılığını göstermektedir.

Kapsama oranı, Tablo 5.4'deki yük tespiti ile karşılaştırıldığında yaklaşık %10 azalsa da, ortalama %83.85 kapsama ile tetikleyici kısmı tespit edebilmekteyiz. Kapsama oranındaki azalma, tetikleyici kısmın devrede herhangi bir yola müdahale etmemesi, bunun yerine sadece bağlı olduğu kapıların fanout'unu artırmasından kaynaklanmaktadır.

Truva atının sadece yük kısmından değil, aynı zamanda tetikleyici kısmından da tespit edilmesi, saldırganlar için kesinlikle caydırıcı olacaktır. Şekil 3.4'deki Truva atını rast-

gele c5315'e yerleřtirmek isteyen bir saldırganı dűőűnelim. %95.45 kapsama olduėundan, yűkű kapsanmayan bir kenara yerleřtirme olasılıėı 0.0455'tir. Tablo 5.6'dan tetikleyicinin giriřinin tespit edilemeyen bir kenara baėlanma olasılıėı 0.0903'tűr. Ancak dűrt giriř vardır ve bunların tűmű tespit edilemeyen kenarlara yerleřtirilmelidir. Bűyle bir Truva atının tespit edilememe olasılıėı řu řekildedir:

Tablo 5.6: Truva Atının Tetikleyici Kısımına ait Sonular.

Test Devreleri	řűpheli Kenarlar	Kapsanan Kenarlar	Kapsanan Kenarlar (%)	Birleřtirilmiř Tespit Olasılıėı (%)
880	507	326	64.89	99,87
1355	856	658	78.27	99,97
1908	1419	1182	82.24	99,99
2670	1850	1577	86.76	99,99
3540	2630	2411	90.53	> 99,99
5315	3878	3554	90.97	> 99,9
6288	4288	3445	82.60	99,98
7552	5836	5533	94.57	> 99,99

$$P(Tespit; Edilemeyen; DTA) = 0.0455 \times 0.0903^4 = 3.025 \times 10^{-6} \quad (5.4)$$

Bu, inanılmaz derecede kűűűk bir olasılıktır, yani Truva atını yűkű veya tetikleyici giriřlerinden birini yakalayarak tespit etme olasılıėımız %99.99'dan fazladır. Tablo 5.4 ve Tablo 5.6 kullanılarak, her devre iin benzer bir birleřik tespit olasılıėı sunulmuřtur ve bu olasılık Tablo 5.6'un son sűtűnűnda gűsterilmiřtir.

5.5. Ardışık Devrelerde Truva atı Tespitine Dair Sonuçlar

Çalışmamız boyunca varsayımımız Truva atlarının üretim aşamasında devreye eklenmesi oldu. Bu nedenle ardışık devre olarak, Trusthub’da [38] bulunan ve Truva atı sınıflandırması yapısına göre, üretim aşamasında eklenen Truva atlarına ayarlanamış olan test devlerini kullandık. Tablo 5.7 da sonuçları gösterilen devreler, iki ayrı test devresi olarak tanımlanmıştır. Fakat aslında Multpyramid adlı ardışık devrenin iki farklı bölgesine aynı tipte Truva atının eklenmesiyle oluşturulmuştur. Her ikisine de Truva atının tetikleyici kısmında bir adet 4 girişli VE kapısı eklenmiştir. Multpyramid-T100 adlı test devresinde Truva atı devrenin orta bölgelerinden birine, Multpyramid-T200 adlı ise devrenin çıkışlarından birine yakındır.

Devrelerin sahip oldukları özelliklere ait bilgiler ilk üç sütunda, Truva atı olarak tetikleyici kısmı VE kapısı, yük kısmı XOR kapısı olan Şekil 3.4’deki yapı eklenmiştir. Bölüm 4.3’de bahsedilen RUY yöntemiyle test ettiğimiz için bir önceki başlık olan Bölüm 5.4 ile aynı şekilde uygulama yapılmıştır. Truva atının tetikleyici kısmının dört adet bacağının devreye bağlandığı her bir kenar şüpheli kabule dilmiştir. Bunlara ek olarak, asıl şüpheli kenar olarak, Truva atının yük kısmının devreye bağlandığı bacakta bulunmaktadır. Böylece şüpheli kenar sayısı 5 olmaktadır. her bir kenar için RUY algoritması çalıştırılarak, birer yol çifti bulunur. Bulunan yol çiftleri üzerinde Bölüm 4.3.4’de anlatılan ve RUY yönteminin parçası kullanılarak Tablo 5.7’in son sütunu olan Truva atı tespit yüzdesi elde edilir.

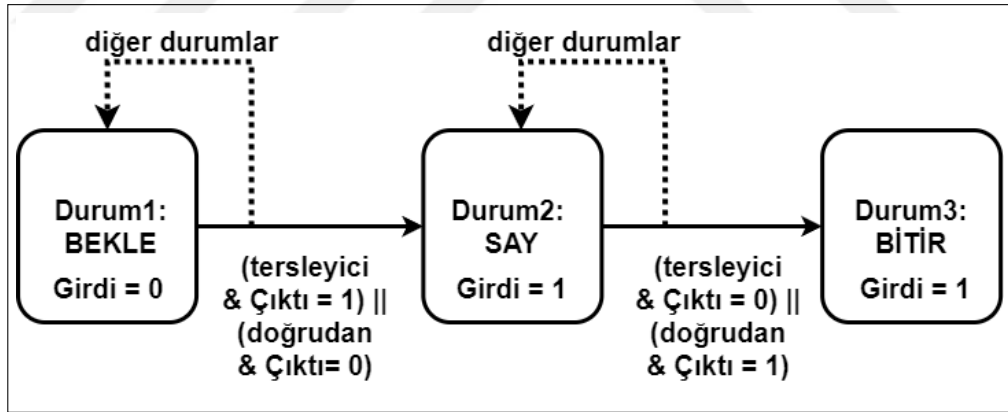
Tablo 5.7: Trust-Hub Ardışık Devrelerine ait Sonuçlar.

Devreler	Girdi Sayısı	Çıktı Sayısı	Kapı Sayısı	Şüpheli Kenarlar	Yol Çiftleri	Tespit Yüzdesi
Multpyramid-T100	43	10	1586	5	5	%100
Multpyramid-T200				5	5	%80

6. Yol Gecikmesi ile Truva Atı Tespitinin FPGA Tabanlı Doğrulaması

6.1. Yol Gecikme Ölçüm Devresi

Bir FPGA içindeki bir yolun gecikmesini hesaplamak için, Şekil 6.1’de gösterilen durum makinesi ile basit bir ardışık devre tasarlanmıştır. Bu devre, test edilmek istenen yolun girişine ve çıkışına bağlanmaktadır. Test edilen yoldaki kapıların yan girişlerinin tümü başlangıçta kontrol etmeyen (ing. non-controlling) değerlere ayarlanmaktadır. Bu değerler, kapının diğer giriş değerlerine bağlı olarak çıkışı etkilemeyen değerlerdir. AND ve NAND kapıları için yan girişlere verilmesi gereken kontrol etmeyen değer 1 iken, OR ve NOR kapıları için bu değer 0’dır. Kontrol etmeyen değerlerin ayarlanması sayesinde, yapılan her giriş değişikliği yolun sonuna kadar, yani çıkışa kadar yayılım sağlar. Bu sayede, çıkışta oluşan değişikliği gözlemleyerek yol gecikmesi ölçümü yapılır.



Şekil 6.1: Yol Gecikmesi Ölçümünü Yapan Sonlu Durum Makinesi Tasarımı.

Şekil 6.1’nin gösterdiği gibi, ilk durumda, test edilen yolun girişine bir lojik-0 uygularız ve çıkışın stabil hale gelmesini bekleriz. Eğer doğrudan (ing. non-inverting) bir yolsa, lojik-0’da stabil hale gelir, tersleyici (ing. inverting) bir yolsa lojik-1’de stabil hale gelir. Daha sonra, ikinci durumda, girişe lojik-1 uygulanır ve bu sırada bir sayaç başlatılır. Son durumda, çıkıştaki sonuç değişikliği beklenmekte ve bu meydana geldiğinde sayaç durdurulmaktadır. Sayaç durduğunda elde edilen değer, giriş değişikliği ile ilgili çıkış

değişikliği arasında oluşan döngü sayısını sağlamaktadır. Bu değeri saat periyodu ile çarptığımızda, test edilen yolun gecikmesini elde etmekteyiz.

Saat hızı, en iyi çözünürlük için test edilen FPGA tarafından izin verilen maksimum hız olarak alınmaktadır. Maksimum izin verilen saat hızının f_{max} ile temsil edildiğini ve sayacın nihai değerinin N olduğunu varsayarsak, elde edilen yol gecikmesi N/f_{max} olmaktadır. Bu tür bir düzenele hesaplanabilecek minimum gecikme farkı $1/f_{max}$ olup, ölçülebilir yolun boyutu için alt bir limit oluşturmaktadır.

Örnek yol olarak, ISCAS'85 test devrelerinden c7552'den bir yol kullanmaktayız [81]. Bu yol, NAND, NOR, AND, OR lojik kapıları ve NOT'lar dahil olmak üzere 35 mantık kapısına sahiptir. Aslında, kapı türlerinin değişimi FPGA üzerindeki sonucu etkilememektedir, çünkü tüm kapı türleri FPGA içinde LUT'lara dönüştürülmektedir.

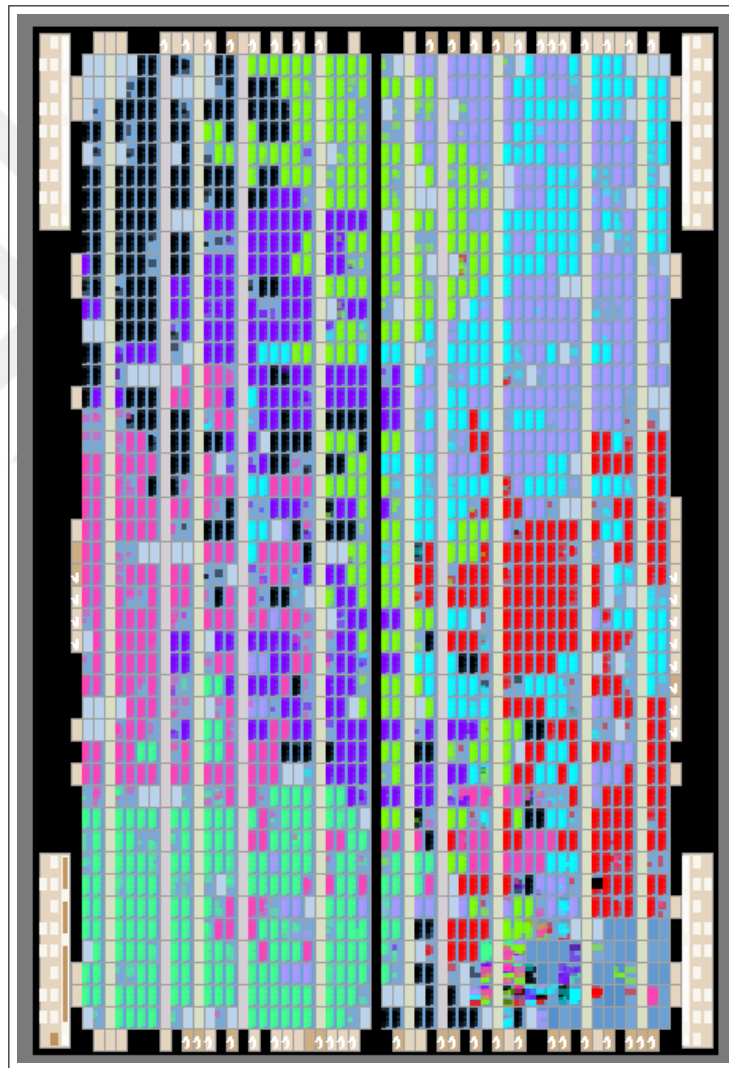
FPGA içine yerleştirirken yolun tüm yan girişlerini kontrol etmeyen değerlere bağlamakta bir sorun bulunmaktadır. FPGA yazılımının optimizasyon araçları, yani Quartus, otomatik olarak optimizasyon uygulamaktadır. Bunun sonucunda, yolu kontrol etmeyen yollar tek bir tamponlayıcıya dönüşmektedir. Yolu tersleyici yollar ise bir inverttere dönüşmektedir. Tüm optimizasyon adımlarını önlemek için, her bir satırda bir giriş veya çıkış tanımlandığında, Verilog'da `/synthesis keep/` yorum satırı olarak eklenmelidir.

Intel Cyclone-V 5CEBA4F23C7 FPGA'larını kullanmaktayız, bu FPGA'lar 28nm proses teknolojisiyle üretilmiştir ve maksimum 250MHz saat hızlarına sahiptir. Seçtiğimiz 35 kapılı yol, ISCAS'85 benchmark'larındaki uzun yollardan biridir. Ancak FPGA 4ns ($1/250MHz$) olan çözünürlüğü, yaklaşık 8ns gecikmesi olan bu yolun gecikmesini kesin olarak ölçmek için çok yüksektir. Yaklaşık olarak yolun gecikmesini ölçebilesek de, (3.4) gösterilen küçük bir Truva atının neden olduğu ek gecikmeyi ölçmek imkansız olacaktır. Hem yolun hem de Truva Atı'nın gecikmesini ölçebilmek için, aynı yolun birden çok kopyasını seri olarak bağlayıp bu seri bağlı yolların gecikmesini ölçmekte ve son olarak toplam gecikmeyi kopya sayısına bölerek tek bir yolun gecikmesini elde etmekteyiz.

6.2. Gecikme Değeri Üzerindeki Süreç Varyasyonu Etkisi

Üretim sürecinin neden olduğu gecikme varyasyonlarını ikiye ayırmış ve Bölüm 1.4'de detaylandırmıştık. Yonga içi gecikme varyasyonu bileşenini ölçmek için, aynı FPGA

ipi iinde aynı yolu 8 farklı konuma yerleřtirmektedir. řekil 6.2'deki her bir renk farklı bir yolu temsil etmektedir. Temsil edilen her yol aslında, aynı yolun 50 kopyasından oluřmaktadır ve bu kopyalar seri olarak baėlanmıřtır. Bylece $250MHz$ saat hızına sahip olan FPGA ile gecikmelerini neredeyse kesin olarak lebilmektedir. Sonuta elde edilen yol gecikmeleri Tablo 6.1'de gsterilmektedir. İlk satır, seri olarak baėlanmış 50 yol kopyasının toplam llen gecikmesidir. İkinci satır ise, tek bir yolun gerek gecikmesini gstermektedir. Bu tablo, yonga ii bileřenden kaynaklanan gecikme varyasyonunun $3\sigma/\mu$ oranının %2.96 olduėunu ortaya koymaktadır.



řekil 6.2: 8 Aynı Yolun FPGA zerinde Farklı Konumlara Yerleřimi.

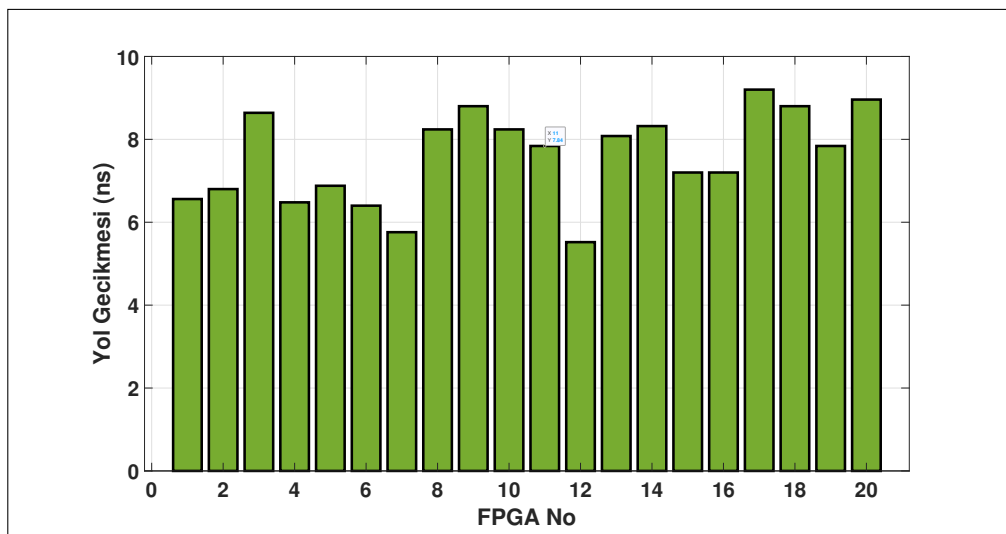
Yonga dıřı varyasyon bileřenini lmek iin, aynı yol farklı FPGA iplerine yklenmekte ve gecikmeler llmektedir. Bunun iin toplamda 20 ip kullanılmaktadır ve sonular

Şekil 6.3'te çubuk grafik olarak gösterilmektedir. Grafikte, Yol gecikmelerinin 5ns ile 10ns arasında değiştiği gözlemlenmiştir. Bu dağılımın standart sapması (σ) ve ortalaması (μ) $3\sigma/\mu$ oranının %43.5 olduğunu göstermektedir, bu oran yonga içi varyasyonlardan çok daha yüksektir. Bu dramatik farkın bir nedeni, bu FPGA'ların farklı levha (ing. wafer) ya da farklı üretim serisinden gelmeleridir. Elimizdeki FPGA'ların aynı waferdan ya da üretim serisinden gelmesini garanti edemiyoruz. Böylece üretilmiş devreler üzerinde süreç varyasyonundan bahsedilirken, sadece yonga-içi ve yongalar arası değil, partiden partiye varyasyon (ing. Lot-to-lot variation) ve levhalar arası varyasyon (ing. wafer-to-wafer variation)'un da dikkate alınması gerekmektedir.

Tablo 6.1: FPGA üzerinde 8 Farklı Konumdaki Yol Gecikmeleri (nano saniye).

ID	L1	L2	L3	L4	L5	L6	L7	L8
50 Yol	428	436	436	440	432	436	428	436
1 Yol	8.56	8.72	8.72	8.8	8.64	8.72	8.56	8.72

Devredeki gecikme artışı, Truva atı veya varyasyonlardan kaynaklanabilir, bu tür gecikme varyasyonları DTA'ların gizlenmesi için oldukça uygun bir alan açmaktadır.

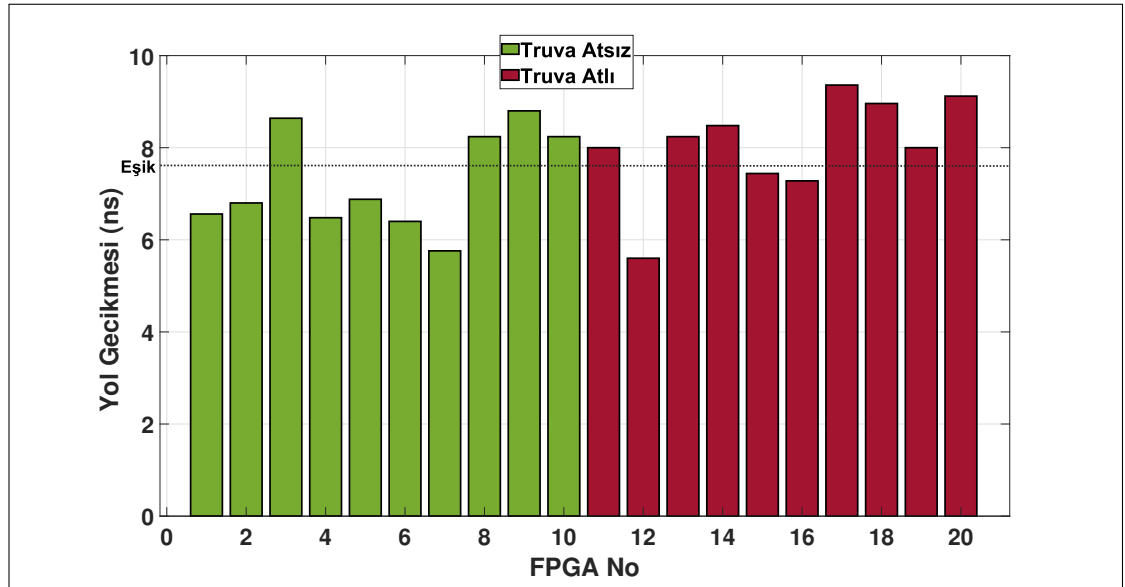


Şekil 6.3: 20 FPGA üzerinde ölçülen yol gecikmeleri.

6.3. Truva Atının Gecikmeye Etkisi

Önceki bölümde, gerçek 28nm FPGA çiplerindeki varyasyonun etkisi gösterilmiştir. Bu bölümde, eklenmiş Truva Atı'nın etkisi incelenmektedir. Bu amaçla, Şekil 6.4'de gösterilen Truva Atı'nın yük kısmı olarak bir XOR kapısı yerleştirilmektedir [16, 38]. Adil sonuçlar elde etmek için, test edilen yolun her kopyasına Truva Atı yerleştirilmiştir. Şekil 6.4, 10 adet Truva atı içermeyen yol gecikmesini ve 10 adet Truva atı yerleştirilmiş yol gecikmesini göstermektedir, her biri farklı bir FPGA'ya karşılık gelmektedir. Bu bölüm altındaki tüm şekiller boyunca, Truva atı içermeyen yongalar yeşil renkle ve Truva atı yerleştirilmiş yongalar kırmızı renkle gösterilmektedir.

Şekil 6.4, Truva atı yerleştirilmiş bir yolun, Truva atı içermeyen bir yoldan çok daha küçük bir gecikmeye sahip olabileceğini göstermektedir. Örneğin, FPGA-3 yaklaşık 9ns Truva atı içermeyen yol gecikmesine sahipken, FPGA-12 daha küçük bir gecikmeye sahip Truva atı yerleştirilmiş yola sahiptir. Bu, varyasyonların Truva atının etkisini mükemmel bir şekilde gizleyebileceğini ve sadece yol gecikmesine bakarak Truva atı tespitinin mümkün olmadığını açıkça göstermektedir.



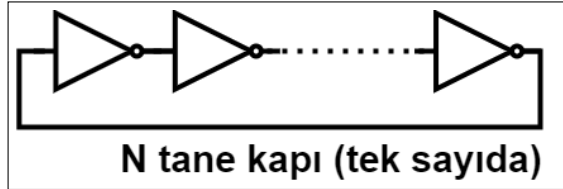
Şekil 6.4: Yarisına Truva Atı Eklenmiş FPGA'ler üzerinde Ölçülen Yol Gecikmeleri.

Truva Atı eklenmiş olanları Truva Atı içermeyenlerden ayırt etmek istersek, Şekil 6.4'te siyah bir çizgi ile gösterilen dikkatlice seçilmiş bir eşik bile, 20 çip arasında 3

yanlış negatif (ing. false negatif) ve 4 yanlış pozitif (ing. false positive) ile sonuçlanmaktadır. Yanlış pozitifler, Truva atı yerleştirildiği halde Truva atı içermeyen olarak etiketlenenlerdir. Yanlış negatifler, Truva atı olduğu halde Truva atı içermeyen olarak etiketlenenleri ifade etmektedir. Ancak, test eden kişi ya da cihaz böyle bir eşiği seçemez, çünkü elinde sadece 20 adet FPGA'ya ait yol gecikmesi sonuçları vardır. Hangisinin Truva atı içerdiğine bir bilgi yoktur. Sadece gecikme değerlerinin dağılımlarına bakarak eşik değeri belirlemesi mümkün olmayacaktır. Bunlara ek olarak, test sırasında tüm yongaların Truva Atı içermediği varsayabilir. Çünkü gecikmeleri birbirine karışmıştır ve Truva Atı'nın etkisi tamamen üretim varyasyonlarının altında gizlenmiştir. Bu sorunu çözmek için varyasyonların etkisi bastırılmalıdır, bu durum sonraki bölümde açıklanmıştır.

6.4. Varyasyonları Bastırmak için Halka Osilatörü Kullanımı

Bir halka osilatörü (ing. Ring Oscillator), tek sayıdaki seri bağlı inverterlerden oluşmakta ve yalnızca çıkışının frekansına bakarak gecikmeyi belirlemek için kullanılmaktadır (Şekil 6.5). Çünkü çıkış değeri, sürekli olarak halka osilatörün gecikmesinin iki katı bir periyot ile tetiklenmektedir.

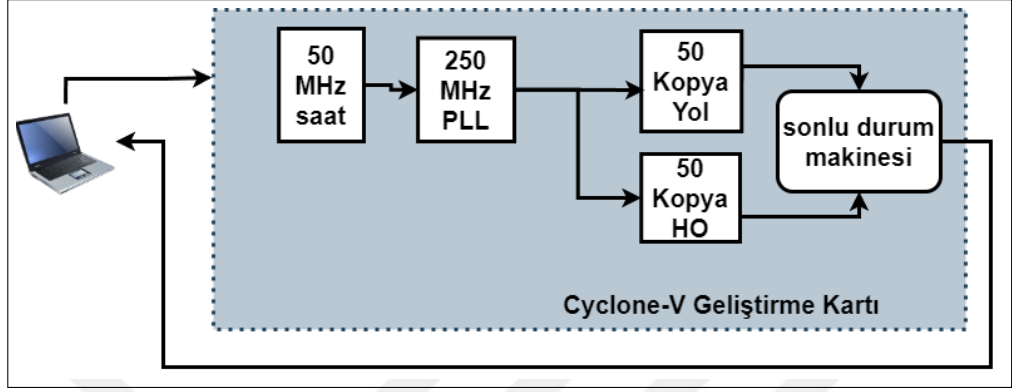


Şekil 6.5: Genel Bir Halka Osilatör Devresi.

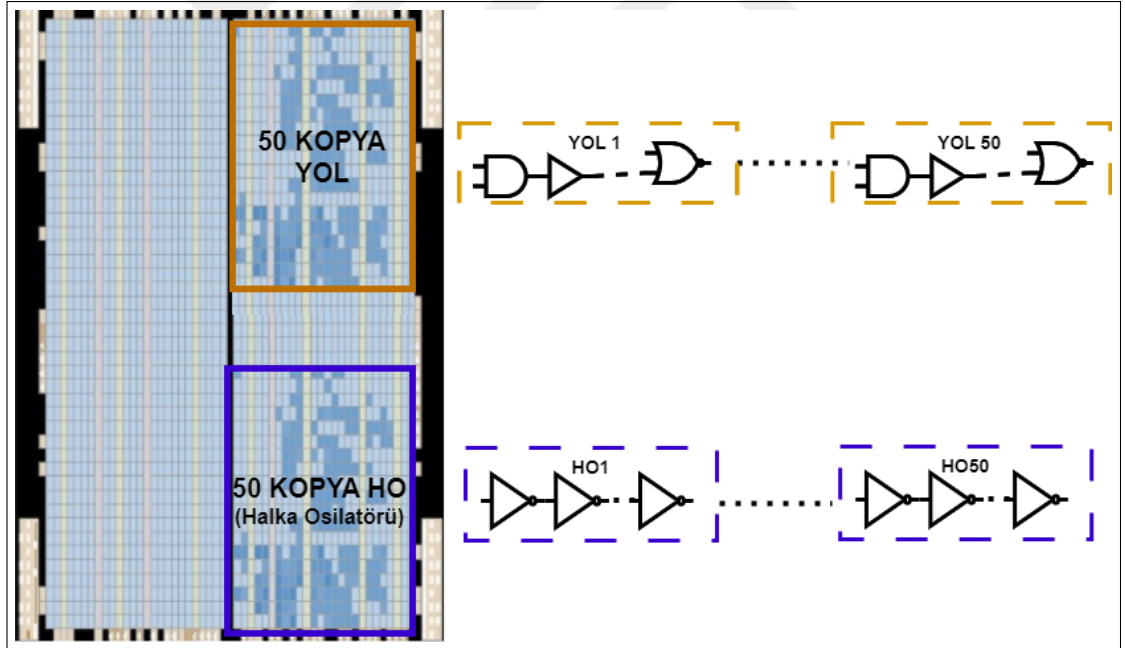
Belirli bir FPGA yongasının gecikmesi hakkında bir fikir vermek için 35 DEĞİL kapısından oluşan Halka Osilatörler kullanılmaktadır. Bu Halka Osilatör (HO), yer aldığı yonganın varyasyonlardan nasıl etkilendiğini temsil etmek için kullanılmaktadır. Bu nedenle, 20 FPGA yongasının her birine ek bir halka osilatör yerleştirip gecikmelerini ölçmekteyiz. Deney düzeneği Şekil 6.6'de gösterilmekte ve devreyi yerleştirdikten sonraki FPGA düzeni Şekil 6.7'de gösterilmektedir.

Halka osilatörü yonga dışı varyasyonlardan benzer şekilde etkilendiği için, her bir yonga için test edilen yolun gecikmesini halka osilatörün gecikmesine bölmek, varyasyonlardan

kurtulmayı sağlayabilir. Bu nedenle, her yonga için test edilen yolun gecikmesini halka osilatörün gecikmesine bölmekteyiz. Elde edilen 20 gecikme oranı Şekil 6.8’da gösterilmektedir.



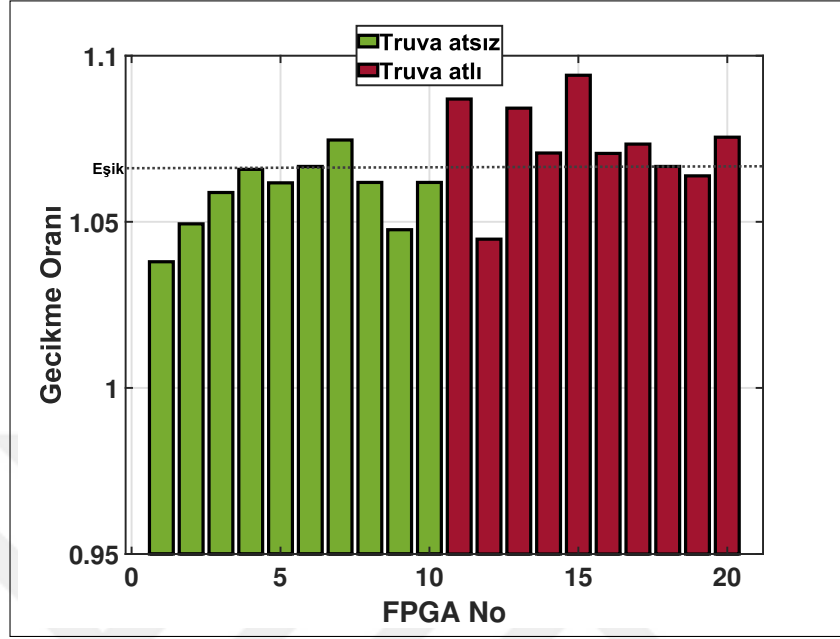
Şekil 6.6: Test Düzenine Genel Görünümü.



Şekil 6.7: 50 Kopya Yol ve HO'nun FPGA Yerleşiminin Gösterimi.

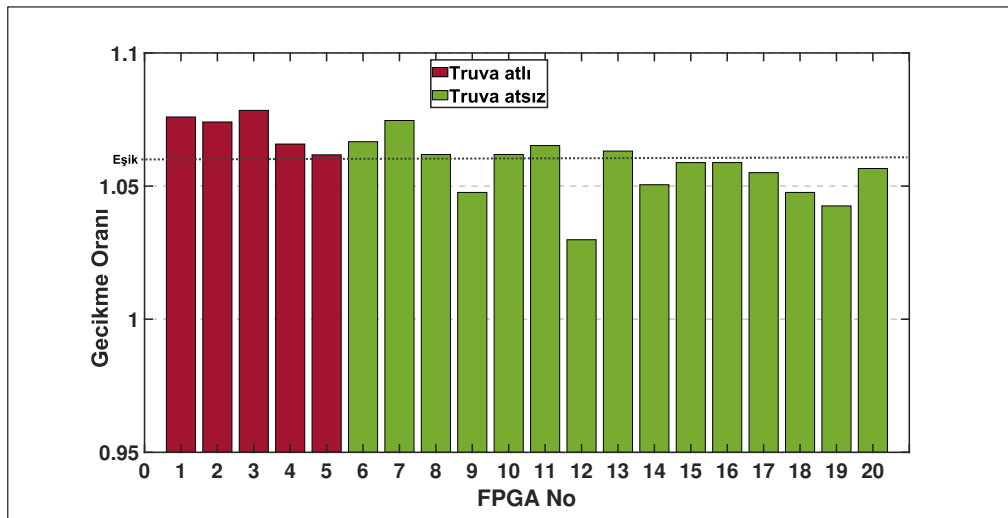
Aslında, Truva atı yerleştirilmiş çiplere karşılık gelen gecikme oranları, Truva atı içermeyen çiplere göre daha büyük olma eğilimindedir. Ancak, çipleri ayırt etmek yine de çok zordur. Örneğin, Şekil 6.8’da siyah bir çizgi ile gösterilen neredeyse mükemmel bir eşik koyarsak, yine de 20 çip arasında 1 yanlış pozitif ve 2 yanlış negatif

elde etmekteyiz, bu da Şekil 6.4 sonuçlarından biraz daha iyidir, Şekil 6.4 sadece yol gecikmesi değerleri kullanılmıştı.



Şekil 6.8: Yarisına Truva Atı Eklenmiş FPGA'ler üzerinde Yol Gecikme Oranları.

Aynı deneyi tekrarlıyoruz, ancak bu sefer son 10 FPGA yerine Truva atımı ilk 5 FPGA'ya yerleştiriyoruz. Elde edilen oran grafiği Şekil 6.9'da gösterilmektedir. Yine de, çok iyi seçilmiş bir eşik değeri bile 6 yanlış pozitif sonuç vermektedir.



Şekil 6.9: Dörtte Birine Truva Atı Eklenmiş FPGA'ler üzerinde Yol Gecikme Oranları.

Ayrıca, gerçek durumda test eden kişi burada renklendirilmiş çubukların renklerini görememektedir. Sadece 20 gecikme oranı hesaplaması toplanabilmektedir. Bu nedenle, Truva atı yerleştirilmiş yongaları Truva atı içermeyenlerden ayırt etmek için kullanılacak eşik değeri bilinmemektedir.

Burada eşik değerini belirlemek için, gecikme oranı ölçümlerinin dağılımının çift model olup olmadığını tespit etme yöntemi [78] kullanılabilir. Eğer Truva atı etkisi yeterince büyükse, Truva atı içermeyen ve Truva atı yerleştirilmiş yongalar, iki farklı merkeze kümelenmiş gecikme oranı dağılımlarına neden olacaktır. Bu dağılımların merkezlerinin farklılaşmış olması, çift model olduğunu göstermektedir. Bunun sonucunda, eğer dağılımlar çift model ise, bazı yongalarda Truva atı yerleştirildiğini söyleyebiliriz.

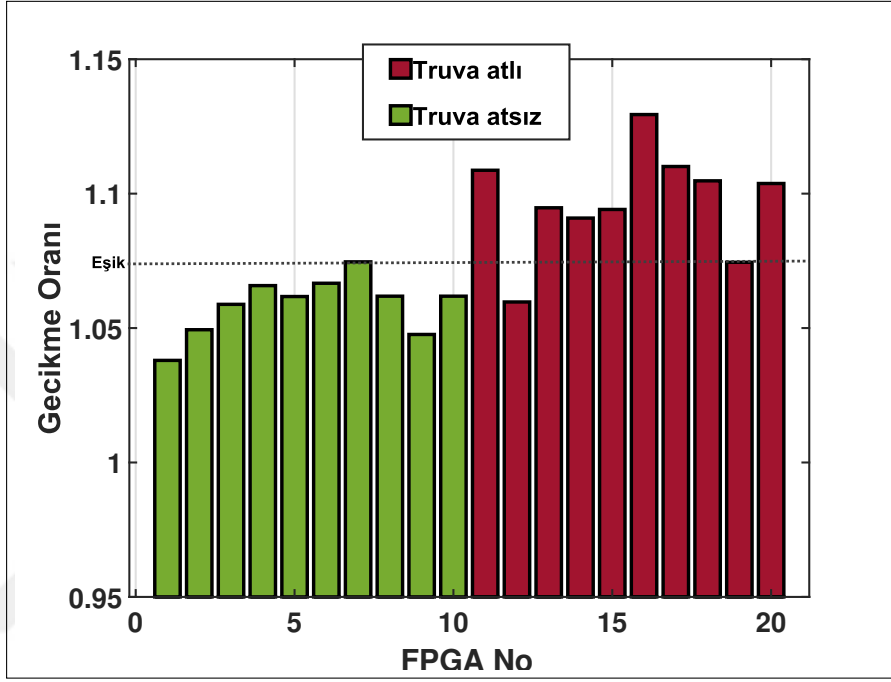
6.5. Truva Atının Boyut Değişimi

Truva Atı'nın boyutu ve varyasyon miktarı bu sonucun ana nedenleridir. Çünkü bu deneylerde Truva Atı'nın boyutu yol boyutuna göre yaklaşık %2.7 olup, yonga içi varyasyonlardan daha küçüktür. Önceki incelemelerden, yonga-içi varyasyonu %2.96, yongalar arası varyasyonu %43.5 olduğunu gözlemlemiştik. Şekil 1.6'den daha büyük bir Truva Atı yükü kullanarak, Truva Atı'nın etkisini arttırabiliriz. Bu sayede çok daha iyi tespit senaryoları ve Hartigan testlerinin sonunda tek model ya da çift model sonuçları sağlanmaktadır.

Örneğin, bir XOR yükü yerine iki seri XOR kullanırsak, elde edilen gecikme oranları Şekil 6.10'da gösterilmektedir. Testin sonucunda, şekilde gösterilen eşik seçilebilirse sonuçlar öncekilere göre iyi olmaktadır. Ancak hala iki yanlış negatif sonuç bulunmaktadır. Öte yandan, seri eklenen XOR sayısını üçe çıkardığımızda, elde edilen gecikme oranları sonunda mutlak bir ayırım sağlayabilmektedir. Burada mutlak ayırımdan kastedilen, gecikme oran dağılımları incelendiğinde iki ayrı merkezde kümelenmiş iki adet dağılım elde edilebilmesidir.

20 yongadan gelen gecikme oranı dağılımlarını kontrol ettiğimizde, 1 XOR, 2 XOR, 3 XOR ve 5 XOR Truva atı durumları için elde edilen histogramlar sırasıyla DTax1, DTax2, DTax3 ve DTax5 olarak Şekil 6.12'te gösterilmektedir. Şekilden çıkarılan sonuç, elde edilen gecikme oranı dağılımının, çift model olma olasılığının Truva atının boyutu arttıkça arttığıdır. Bu sonuç Hartigan sonuçları ile de doğrulanmaktadır. Çünkü

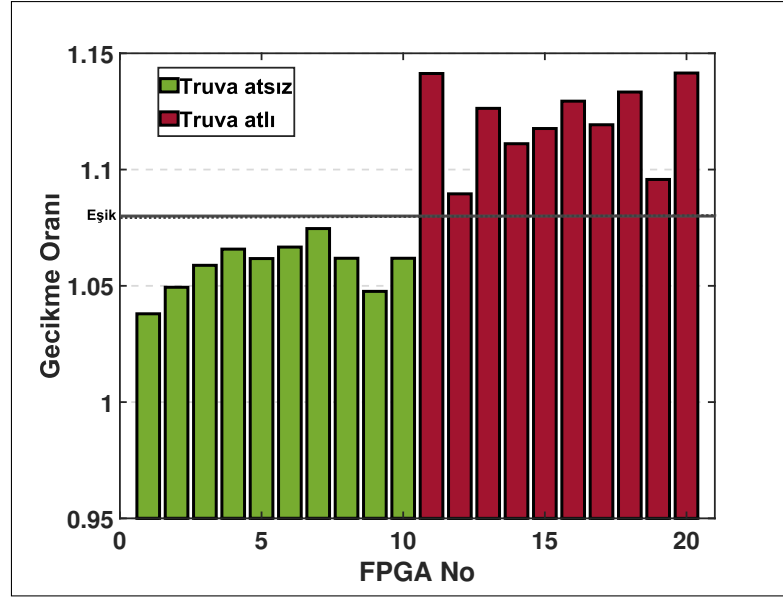
Hartigan testinin verdiği p değerleri Tablo 6.2’de gösterilmiştir, burada 0.1’den küçük p değerleri dağılımın çift model olduğunu göstermektedir [46]. Tablodan elde edilen sonuç, DTax5 durumu için çift model tespiti yapılmakta ve dolayısıyla Truva atı tespit edilmektedir. Böylece, Truva Atı yerleştirilmiş devreleri ayırt etmek için iki dağılımın ortasına ait bir eşik kolayca bulunabilmektedir.



Şekil 6.10: Truva Atı Yük Kısmı: İki Adet Seri Bağlı XOR Kapısı.

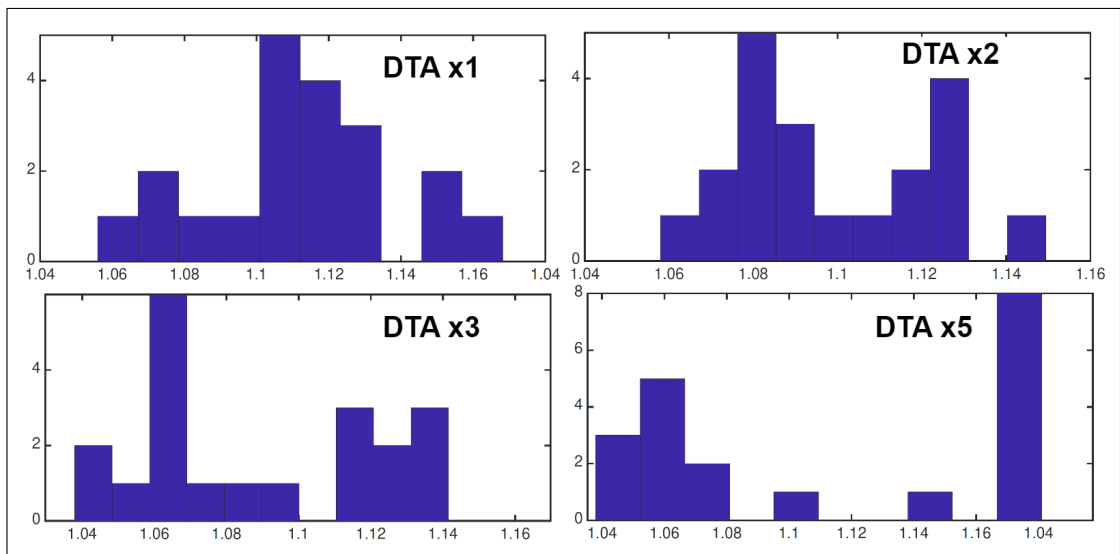
Literatürdeki başka bir çift model testi, Warren Sarle’nin testidir [82], bu testte 0.55’ten büyük katsayılar çift model veya çoklu model dağılım anlamına gelirken, bu değerden düşük olanlar tek model dağılım anlamına gelmektedir. Şekil 6.12’teki dağılımlar Sarle’nin testine verildiğinde, elde edilen sonuçlar Tablo 6.2’de gösterilmektedir. Hartigan Dip testine benzer şekilde, sonuçlar Truva atı boyutu arttıkça çift model dağılıma daha yakın hale gelmektedir, ancak sadece DTax5 durumu çift model olarak etiketlenebilmiştir.

20 FPGA yongasında yapılan ölçümlerden, gecikme varyasyonlarının etkisinin oldukça ezici olduğunu gösterdik. Tablo 6.2’den çıkardığımız sonuç; çift model testine dair her iki yöntemde de sadece DTax5 boyutu için çift model olduğu tespit edilebilmiştir. Diğer Truva atı boyutları tek model olarak tespit edilmiştir.



Şekil 6.11: Truva Atı Yük Kısmı: Üç Adet Seri Bağlı XOR Kapısı.

DTAx5 boyutuna kadar yapılan teslerde, varyasyonların karakteristiğini yakalamak için devreye ek bir halka osilatör yerleştirsek bile, elde edilen gecikme oranları Hartigan testinden unimodal dağılım olarak geçmektedir. Bu da sonuçta elde edilen gecikmelerin tek bir merkezde kümелendiği ve yerleştirilen Truva atlarının tespit edilmediği anlamına gelmektedir. Sonuç olarak DTAx5 boyutundan daha küçük bir truva atının her iki yöntemle tespit edilmesi mümkün görünmemektedir. DTx5 boyutundan küçük olan örneklerde ise, varyasyon etkisi DTA etkisi baskılamaktadır.



Şekil 6.12: Farklı Truva Atı Boyutları için Gecikme Oranı Histogramları.

Tespit şeması, Truva atı boyutu arttıkça daha iyi hale gelmektedir. Ancak, saldırganlar, Truva atını gizlemek ve oluşturmak için büyük yükler kullanmak zorunda değildir. Tek bir XOR kapısının (DTAx1) yükü bile, Truva atını mantık testlerinden gizlemek ve devrenin işlevselliğini değiştirmek için yeterlidir. Bu, varyasyonları bastırmak ve devredeki Truva atının etkisini ortaya çıkarmak için sofistike gecikme tabanlı tespit yöntemlerinin gerekliliğini göstermektedir.

Tablo 6.2: Çift Model Yöntemlerinin Test Sonuçları.

Bimodality Test	DTAx1	DTAx2	DTAx3	DTAx5
Hartigan Yöntemi	0.61	0.35	0.21	0
Sarle Yöntemi	0.29	0.42	0.52	0.66

7. Tartışmalar

Bu tezde, donanımsal Truva atlarının tespiti için yenilikçi yöntemler geliştirilmiş ve çeşitli deneysel sonuçlar ile bu yöntemlerin etkinliği gösterilmiştir. Elde edilen bulgular, aşağıdaki tartışma konularını gündeme getirmektedir:

- **Yöntemin Etkinliği ve Sınırları:** Geliştirilen yöntemler, belirli Truva atı modelleri ve süreç varyasyonları altında yüksek başarı oranlarına ulaşmıştır. Ancak, bu yöntemlerin daha karmaşık devreler ve çeşitli DTA modellerinde nasıl performans göstereceği tartışılmalıdır. Devre karmaşıklığı arttıkça devredeki kenar sayısı artmaktadır. Bu da devredeki yol sayısını doğrudan etkilemektedir. Devredeki yol sayısı arttıkça, 8x8 ızgara modeline göre, bir ızgaradan geçen kapı sayısının da artacağı varyılabilir. Sonuç olarak, devre karmaşıklığı arttıkça Truva atı tespiti için yol çifti bulmak sorun olmayacaktır. Tüm devre kenarları için değil sadece şüpheli bölge/kenarlar için yöntemimizi çalıştırmak, çalışma zamanı açısından oluşan maliyeti de önleyebilir. Aynı benzerlik çeşitli DTA modelleri içinde söylenebilir. Truva atı modelimiz, literatürdeki gecikme tabanlı yöntemlerde kullanılan en küçük boyutlu Truva atıdır. Bundan daha büyük boyutlu her Truva atı, yol gecikmesine daha fazla etki edecektir.
- **Süreç Varyasyonların Modellenmesi:** Süreç varyasyonlarının donanım güvenliği üzerindeki etkileri büyük önem taşımaktadır. Bu nedenle, kullanılan varyasyon modelini güncel teknolojiye yakınlaştırmak için yapılması gerekenler tartışılabilir.

7.1. Gelecek Çalışmalar

Bu tezde sunulan bulgular, donanım güvenliği alanında gelecekte yapılacak çalışmalar için bir temel oluşturmaktadır.

- **Daha Geniş Ölçekli Devrelerin İncelenmesi:** Sunulan yöntemlerin daha büyük ve karmaşık entegre devrelerde uygulanabilirliğini araştırmak, yöntemlerin genelleştirilebilirliğini ve etkinliğini artırmak için önemlidir. Daha geniş ölçekli ve sıralı (ing. sequential) devrelerde yapılacak testler, yöntemlerin performansını ve sınırlarını daha iyi anlamamızı sağlayacaktır.

- **Yöntemlerin Donanım Uygulamaları ile Entegrasyonu:** Geliştirilen yöntemlerin FPGA ve ASIC gibi donanım platformlarına entegrasyonu ve bu platformlarda gerçek zamanlı testlerin yapılması, yöntemlerin pratik uygulanabilirliğini gösterecektir. Bu entegrasyon çalışmaları, yöntemlerin endüstriyel kullanıma uygunluğunu değerlendirmek için kritik öneme sahiptir.
- **Makine Öğrenmesi ve Yapay Zeka Tekniklerinin Kullanımı:** Makine öğrenmesi ve yapay zeka teknikleri, Truva atı tespitinde önemli bir potansiyele sahiptir. Geçtiğimiz beş yıl içinde özellikle çalışma zamanı sırasında Truva atı tespit eden ve tersine mühendislik çalışmaları yapılan alanlarda oldukça fazla sayıda yayın ve araştırma sunulmuştur.



KAYNAKLAR

- [1] Francq, J. (2014), “Hardware Trojans: Taxonomy and Detection Methods”, <https://www.cs.ucr.edu/nael/ee260/lectures/hardware-trojans.pdfpage=10.00> , Accessed: 2024-08-18.
- [2] Palumbo, A. (2020), “Introduction to Hardware Security”, https://atc.mise.gov.it/images/documenti/seminari/ottavi_palumbo_hardware_security.pdf , Accessed: 2024-08-18.
- [3] Pazmino, L. (2023), “Real Men Have Fabs”, <https://luisepazmino.substack.com/p/real-men-have-fabs> , Accessed: 2024-08-18.
- [4] Wolff, F. Papachristou, C. Bhunia, S. Chakraborty, R. S. (2008), Towards trojan-free trusted ICs: problem analysis and detection scheme, in “Proceedings of the conference on Design, automation and test in Europe”, ACM, Munich Germany, 1362–1365.
- [5] Li, H. Liu, Q. Zhang, J. (2016), “A survey of hardware Trojan threat and defense”, *Integration*, 55, 426–437.
- [6] Xue, M. Gu, C. Liu, W. Yu, S. O’Neill, M. (2020), “Ten years of hardware Trojans: a survey from the attacker’s perspective”, *IET Computers & Digital Techniques*, 14(6), 231–246.
- [7] Salmani, H. Tehranipoor, M. Karri, R. (2013), On design vulnerability analysis and trust benchmarks development, in “2013 IEEE 31st International Conference on Computer Design (ICCD)”, IEEE, Asheville, NC, USA, 471–474.
- [8] Sinanoglu, O. (2016), Do you trust your chip?, in “2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS)”, IEEE, Istanbul, Turkey, 1–1.
- [9] Xiao, K. Forte, D. Jin, Y. Karri, R. Bhunia, S. Tehranipoor, M. (2017), “Hardware Trojans: Lessons Learned after One Decade of Research”, *ACM Transactions on Design Automation of Electronic Systems*, 22(1), 1–23.
- [10] Rangarajan, N. Patnaik, S. Knechtel, J. Rakheja, S. Sinanoglu, O. (2021), “The next era in hardware security: a perspective on emerging technologies for secure electronics”, Springer, Cham, Switzerland.
- [11] Trippel, T. Shin, K. G. Bush, K. B. Hicks, M. (2020), ICAS: an Extensible Framework for Estimating the Susceptibility of IC Layouts to Additive Trojans, in “2020 IEEE Symposium on Security and Privacy (SP)”, IEEE, San Francisco, CA, USA, 1742–1759.
- [12] Tehranipoor, M. Koushanfar, F. (2010), “A Survey of Hardware Trojan Taxonomy and Detection”, *IEEE Design & Test of Computers*, 27(1), 10–25.
- [13] Bhunia, S. Hsiao, M. S. Banga, M. Narasimhan, S. (2014), “Hardware Trojan Attacks: Threat Analysis and Countermeasures”, *Proceedings of the IEEE*, 102(8), 1229–1247.

- [14] Yang, Y. Ye, J. Cao, Y. Zhang, J. Li, X. Li, H. Hu, Y. (2020), Survey: Hardware Trojan Detection for Netlist, in “2020 IEEE 29th Asian Test Symposium (ATS)”, IEEE, Penang, Malaysia, 1–6.
- [15] Narasimhan, S. Du, D. Chakraborty, R. S. Paul, S. Wolff, F. G. Papachristou, C. A. Roy, K. Bhunia, S. (2013), “Hardware Trojan Detection by Multiple-Parameter Side-Channel Analysis”, *IEEE Transactions on Computers*, 62(11), 2183–2195.
- [16] Cui, X. Koopahi, E. Wu, K. Karri, R. (2018), “Hardware Trojan Detection Using the Order of Path Delay”, *ACM Journal on Emerging Technologies in Computing Systems*, 14(3), 1–23.
- [17] Cha, B. Gupta, S. K. (2013), Trojan Detection via Delay Measurements: A New Approach to Select Paths and Vectors to Maximize Effectiveness and Minimize Cost, in “Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013”, IEEE Conference Publications, Grenoble, France, 1265–1270.
- [18] Yoshimizu, N. (2014), Hardware trojan detection by symmetry breaking in path delays, in “2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)”, IEEE, Arlington, VA, USA, 107–111.
- [19] Vakil, A. Mirzaeian, A. Homayoun, H. Karimi, N. Sasan, A. (2021), “AVATAR: NN-Assisted Variation Aware Timing Analysis and Reporting for Hardware Trojan Detection”, *IEEE Access*, 9, 92881–92900.
- [20] Hoque, T. Narasimhan, S. Wang, X. Mal-Sarkar, S. Bhunia, S. (2017), “Golden-Free Hardware Trojan Detection with High Sensitivity Under Process Noise”, *Journal of Electronic Testing*, 33(1), 107–124.
- [21] Liu, Y. Huang, K. Makris, Y. (2014), Hardware Trojan Detection through Golden Chip-Free Statistical Side-Channel Fingerprinting, in “Proceedings of the 51st Annual Design Automation Conference”, ACM, San Francisco CA USA, 1–6.
- [22] TAMU, (2015), “VLSI Computer-Aided Design (CAD) and Test Research Group, Layout design”, <http://dropzone.tamu.edu/> , Accessed: 2024-08-18.
- [23] Holger, V. (2021), “Ngspice User’s Manual Version 34 (ngspice release version)”, <https://trust-hub.org/benchmarks/chip-level-trojan> , Accessed: 2024-08-18.
- [24] Huffmire, T. Irvine, C. Nguyen, T. D. Levin, T. Kastner, R. Sherwood, T. (n.d.), “Handbook of FPGA design security”, Springer Science & Business Media.
- [25] Skorobogatov, S. Woods, C. (2012), Breakthrough silicon scanning discovers backdoor in military chip, Leuven, Belgium,, 23–40.
- [26] Ullrich, J. (2013), “Barracuda: Back Door”, <http://isc.sans.edu/diary/Barracuda+Back+Door+/15004> , Accessed: 2024-08-18.
- [27] Rogers, M. Ruppertsberger, S. (2012), “Investigative Report on the U.S. National Security Issues Posed by Chinese Telecommunications Companies Huawei and ZTE”.
- [28] Robertson, J. Riley, M. (2012), “The Big Hack: How China Used a Tiny Chip to Infiltrate U.S. Companies”, <https://www.bloomberg.com/news/features/2018->

10-04/the-big-hack-how-china-used-a-tiny-chip-to-infiltrate-america-s-top-companies , Accessed: 2024-08-18.

- [29] Elinoff, G. (2020), “Bitstream Breach: Critical Security Flaw Discovered in FPGA Chips”, <https://www.allaboutcircuits.com/news/bitstream-breach-critical-security-flaw-discovered-in-fpga-chips/> , Accessed: 2024-07-18.
- [30] Kocher, P. Horn, J. Fogh, A. Genkin, D. Gruss, D. Haas, W. Hamburg, M. Lipp, M. Mangard, S. Prescher, T. Schwarz, M. Yarom, Y. (2019), Spectre Attacks: Exploiting Speculative Execution, in “2019 IEEE Symposium on Security and Privacy (SP)”, 1–19.
- [31] Yasaei, R. Faezi, S. Al Faruque, M. A. (2022), “Golden Reference-Free Hardware Trojan Localization Using Graph Convolutional Network”, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 30(10), 1401–1411.
- [32] Hassan, R. Meng, X. Basu, K. Dinakarrao, S. M. P. (2023), “Circuit Topology-Aware Vaccination-Based Hardware Trojan Detection”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42(9), 2852–2862.
- [33] Yang, S. Hoque, T. Chakraborty, P. Bhunia, S. (2022a), “Golden-Free Hardware Trojan Detection Using Self-Referencing”, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 30(3), 325–338.
- [34] Yang, S. Hoque, T. Chakraborty, P. Bhunia, S. (2022b), “Golden-Free Hardware Trojan Detection Using Self-Referencing”, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 30(3), 325–338.
- [35] Jie Li, Lach, J. (2008), At-speed delay characterization for IC authentication and Trojan Horse detection, in “2008 IEEE International Workshop on Hardware-Oriented Security and Trust”, IEEE, Anaheim, CA, USA, 8–14.
- [36] Rai, D. Lach, J. (2009), Performance of delay-based Trojan detection techniques under parameter variations, in “2009 IEEE International Workshop on Hardware-Oriented Security and Trust”, IEEE, San Francisco, CA, USA, 58–65.
- [37] Hoque, T. (2015), Ring Oscillator Based Hardware Trojan Detection, PhD thesis, University of Toledo.
- [38] Salmani, H. (2017), “Trust-HUB Chip level trojan”, <https://trust-hub.org/> , Accessed: 2024-08-18.
- [39] “The Semiconductor Industry’s Biggest Challenges” (2023), <https://www.linkedin.com/pulse/semiconductor-industrys-biggest-challenges-value-driven-solutions/> , Accessed: 2024-08-18.
- [40] Association, S. I. (2011), “International Technology Roadmap for Semiconductors (ITRS) Report 2011 Edition”, <https://trust-hub.org/benchmarks/chip-level-trojan> , Accessed: 2024-08-18.
- [41] Association, S. I. (2023), “International Roadmap for Device and Systems (IRDS 2023) Yield Enhancement”, <https://irds.ieee.org/editions/2023/irds>
- [42] MIT, (2008), “Control of Manufacturing Process”, <https://ocw.mit.edu/courses/2-830j-control-of-manufacturing-processes-sma-6303-spring->

2008/b5882a96aacc8d8a7287b818c2009735_lecture2.pdf , Accessed: 2024-08-18.

- [43] Hongliang Chang, Sapatnekar, S. (2005), “Statistical timing analysis under spatial correlations”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(9), 1467–1482.
- [44] Bayrakci, A. A. (2015), “Stochastic logical effort as a variation aware delay model to estimate timing yield”, *Integration*, 48, 101–108.
- [45] Luo, H. Li, R. Miao, X. Wang, X. (2023), “A comprehensive study of device variability of sub-5 nm nanosheet transistors and interplay with quantum confinement variation”, *Science China Information Sciences*, 66(2), 129402.
- [46] Esirci, F. N. Bayrakci, A. A. (2023), “Delay based hardware Trojan detection exploiting spatial correlations to suppress variations”, *Integration*, 91, 107–118. Publisher: Elsevier.
- [47] Cybersecurity Research, F. (2022), “REFICS: A Step Towards Linking Vision with Hardware Assurance”, https://openaccess.thecvf.com/content/WACV2022/supplemental/Wilson_REFICS_A_Step_WACV_2022_supplemental.pdf , Accessed: 2024-08-18.
- [48] Rajendran, S. Regeena, M. L. (2022), “A Novel Algorithm for Hardware Trojan Detection Through Reverse Engineering”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(4), 1154–1166.
- [49] Bao, C. Forte, D. Srivastava, A. (2016), “On Reverse Engineering-Based Hardware Trojan Detection”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(1), 49–57.
- [50] Botero, U. J. Wilson, R. Lu, H. Rahman, M. T. Mallaiyan, M. A. Ganji, F. Asadizanjani, N. Tehranipoor, M. M. Woodard, D. L. Forte, D. (2021), “Hardware Trust and Assurance through Reverse Engineering: A Tutorial and Outlook from Image Analysis and Machine Learning Perspectives”, *ACM Journal on Emerging Technologies in Computing Systems*, 17(4), 1–53.
- [51] Eggersgluß, S. Milewski, S. Rajski, J. Tyszer, J. (2023), “A New Static Compaction of Deterministic Test Sets”, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 31(4), 411–420.
- [52] Zhang, J. Yuan, F. Wei, L. Sun, Z. Xu, Q. (2013), VeriTrust: verification for hardware trust, in “*Proceedings of the 50th Annual Design Automation Conference*”, ACM, Austin Texas, 1–8.
- [53] Huang, Y. Bhunia, S. Mishra, P. (2016), MERS: Statistical Test Generation for Side-Channel Analysis based Trojan Detection, in “*Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*”, ACM, Vienna Austria, 130–141.
- [54] Cruz, J. Farahmandi, F. Ahmed, A. Mishra, P. (2018), Hardware Trojan Detection Using ATPG and Model Checking, in “*2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID)*”, IEEE, Pune, India, 91–96.

- [55] He, J. Zhao, Y. Guo, X. Jin, Y. (2017), “Hardware Trojan Detection Through Chip-Free Electromagnetic Side-Channel Statistical Analysis”, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(10), 2939–2948.
- [56] Kutzner, S. Poschmann, A. Y. Stöttinger, M. (2013), Hardware trojan design and detection: a practical evaluation, in “Proceedings of the Workshop on Embedded Systems Security”, ACM, Montreal Quebec Canada, 1–9.
- [57] Kocabas, M. Bayrakci, A. A. (2019), Quantification of Power Based Hardware Trojan Detection Under Realistic Variations and Separation of Power Supplies, in “2019 11th International Conference on Electrical and Electronics Engineering (ELECO)”, IEEE, Bursa, Turkey, 522–526.
- [58] S, D. R, R. S. M, N. D. (2021), Hardware Trojan Detection using Ring Oscillator, in “2021 6th International Conference on Communication and Electronics Systems (ICCES)”, IEEE, Coimbatre, India, 362–368.
- [59] Fournaris, A. P. Pyrgas, L. Kitsos, P. (2019), “An efficient multi-parameter approach for FPGA hardware Trojan detection”, *Microprocessors and Microsystems*, 71, 102863.
- [60] He, J. Guo, X. Ma, H. Liu, Y. Zhao, Y. Jin, Y. (2020), Runtime Trust Evaluation and Hardware Trojan Detection Using On-Chip EM Sensors, in “2020 57th ACM/IEEE Design Automation Conference (DAC)”, IEEE, San Francisco, CA, USA, 1–6.
- [61] Faezi, S. Yasaei, R. Al Faruque, M. A. (2021), HTnet: Transfer Learning for Golden Chip-Free Hardware Trojan Detection, in “2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)”, IEEE, Grenoble, France, 1484–1489.
- [62] Wang, H. Panoff, M. Zhan, Z. Wang, S. Bobda, C. Forte, D. (2024), “Programmable EM Sensor Array for Golden-Model Free Run-time Trojan Detection and Localization”, <https://arxiv.org/abs/2401.12193> , Accessed: 2024-06-15.
- [63] Bao, C. Forte, D. Srivastava, A. (2015), “Temperature Tracking: Toward Robust Run-Time Detection of Hardware Trojans”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(10), 1577–1585.
- [64] Blaauw, D. Chopra, K. Srivastava, A. Scheffer, L. (2008), “Statistical Timing Analysis: From Basic Principles to State of the Art”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(4), 589–607.
- [65] Sabri, M. Shabani, A. Alizadeh, B. (2021), “SAT-Based Integrated Hardware Trojan Detection and Localization Approach Through Path-Delay Analysis”, *IEEE Transactions on Circuits and Systems II: Express Briefs*, 68(8), 2850–2854.
- [66] University, N. C. S. (2008), “NanGate45nmOCL”, <https://si2.org/open-cell-library/> , Accessed: 2024-08-18.
- [67] Yier Jin, Makris, Y. (2008), Hardware Trojan detection using path delay fingerprint, in “2008 IEEE International Workshop on Hardware-Oriented Security and Trust”, IEEE, Anaheim, CA, USA, 51–57.

- [68] Sutherland, I. (1999), “Logical effort: designing fast CMOS circuits”, Morgan Kaufmann.
- [69] Esmaeili Taheri, H. Mirhassani, M. (2022), “A Pre-Activation, Golden IC Free, Hardware Trojan Detection Approach”, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 30(3), 315–324.
- [70] Brglez, F. (1989), , Accessed: 2024-08-18.
- [71] Assurance, C. (2020), “Tools and Resources for Hardware Security”, <https://cad-forassurance.org/benchmarks/> , Accessed: 2024-08-18.
- [72] Agarwal, A. Blaauw, D. Zolotov, V. (2003), Statistical timing analysis for intra-die process variations with spatial correlations, in “*ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No.03CH37486)*”, IEEE, San Jose, CA, USA, 900–907.
- [73] Cong, J. Gupta, P. Lee, J. (2010), “Evaluating Statistical Power Optimization”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(11), 1750–1762.
- [74] Sylvester, D. Agarwal, K. Shah, S. (2008), “Variability in nanometer CMOS: Impact, analysis, and minimization”, *Integration*, 41(3), 319–339.
- [75] Esirci, F. N. Bayrakci, A. A. (2020), Acceleration of Spatial Correlation Based Hardware Trojan Detection Using Shared Grids Ratio, in D. Slamanig E. Tsigaridas Z. Zafeirakopoulos, eds, “*Mathematical Aspects of Computer and Information Sciences*”, Vol. 11989, 187–201. Series Title: *Lecture Notes in Computer Science*.
- [76] Lovasz, L. (1993), “Random walks on graphs”, *Combinatorics*, 2, 1–46.
- [77] Chvatal, V. (1979), “A Greedy Heuristic for the Set-Covering Problem”, *Mathematics of Operations Research*, 4(3), 233–235.
- [78] Hartigan, J. A. Hartigan, P. M. (1985), “The Dip Test of Unimodality”, *The Annals of Statistics*, 13(1), 70–84.
- [79] TAMU, (2023), “45nm Layout Files generated by VLSI CAD Group”, <https://github.com/Esirci/ISCAS85-45nm> , Accessed: 2024-08-18.
- [80] Adee, S. (2008), “The Hunt For The Kill Switch”, *IEEE Spectrum*, 45(5), 34–39.
- [81] Brglez, F. Fujiwara, H. (1986), “A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator into FORTRAN”, https://www.researchgate.net/publication/273775783_A_neutral_netlist_of_10_combinational_-_benchmark_circuits_and_a_targeted_translator_in_FORTRAN , Accessed: 2024-08-18.
- [82] Freeman, J. B. Dale, R. (2013), “Assessing bimodality to detect the presence of a dual cognitive process”, *Behavior Research Methods*, 45(1), 83–97.

ÖZGEÇMİŞ

Fatma Nur Esirci, 2015 yılında Gebze Yüksek Teknoloji Enstitüsü Bilgisayar Mühendisliği bölümünden mezun olmuş. Hemen ardından Gebze Teknik Üniversitesi (GTÜ) Bilgisayar Mühendisliği'nde yüksek lisans öğrenimine başlamıştır. Yüksek lisans eğitiminin başında TÜBİTAK proje bursiyeri olarak çalışmıştır. 2017 yılından itibaren aynı kurum olan GTÜ'de araştırma görevlisi olarak çalışmaktadır. Yüksek lisans eğitimi sırasında TÜBİTAK 2210-C yüksek lisans öğrencileri için öncelikli alanlar bursiyeri olarak desteklenmiştir.

Yüksek lisans öğrenimini 2018 yılında tamamlamış, ardından doktora öğrenimine aynı kurumda devam etmiştir. 2021 yılında girdiği doktora yeterlilik sınavının ardından, TÜBİTAK 2211-C doktora öğrencileri için öncelikli alanlar bursiyeri desteklenmeye hak kazanmıştır. Bu tez ile birlikte GTÜ Bilgisayar Mühendisliği bölümünde devam ettiği doktora eğitimi de sonlanmış bulunmaktadır.

TEZ ÇALIŞMASI KAPSAMINDA YAPILAN YAYINLAR

Esirci, F. N., Bayrakci, A. A., (2024), "Hardware Trojan Detection using Path Delays with Spatial Correlation", 20th ACACES Summer School Poster Forum, Fiuggi, Italy, Turkey, 14-20 July.

Esirci, F. N., Mutlu, H., Bayrakci, A. A., (2023), "FPGA Based Verification for the Difficulty of Delay Based Hardware Trojan Detection", 14th International Conference on Electrical and Electronics Engineering (ELECO), Bursa, Turkey, 30 November - 2 December.

Esirci, F. N., Bayrakci, A. A., (2023), "Delay based hardware Trojan detection exploiting spatial correlations to suppress variations", Integration, 91(2023), 107-118. doi: 10.1016/j.vlsi.2023.03.006

Esirci, F.N., Bayrakci, A.A., (2020), "Acceleration of Spatial Correlation Based Hardware Trojan Detection Using Shared Grids Ratio", In Mathematical Aspects of Computer and Information Science s(MACIS), 187-201, Gebze, Turkey, 13–15 November. doi: 10.1007/978-3-030-43120-4