Mikroişlemci Tabanlı Bir Sisteme Hata Enjekte Etme Yöntemi Geliştirilmesi ve Hata Tespit Mekanizmasının Gerçeklenmesi

Improvement Fault Injection Method In A Microprocessor Based System and Implementation Error Detection Mechanism

Buse Ustaoğlu[1], Berna Örs Yalçın[2]

[1]Elektronik ve Haberleşme Mühendisliği Bölümü
İstanbul Teknik Üniversitesi
ustaoglubu@itu.edu.tr

[2]Elektronik ve Haberleşme Mühendisliği Bölümü
İstanbul Teknik Üniversitesi
siddika.ors@itu.edu.tr

Özet

Sayısal sistemlerin güvenilir olarak gerçeklenebilmesi için test edilebilirlik kapasitesinin belirlenmesi gerekmektedir. Özellikle büyük sistemlerin içerisinde kullanılan mikroişlemcilerde oluşabilecek bir hata tüm sistemi etkileyeceğinden testlerinin yapılıp hatalarının tespit edilmesi son derece önemlidir. Bu makalede Sahada Programlanabilir Kapı Dizisi (Field Programmable Gate Array-FPGA) ile gerçeklenmiş bir mikroişlemci ile hata enjekte yöntemi temel alınarak geliştirilmiş bir devre birleştirilerek sistem haline getirilmiştir. Hata üretim devresinden üretilen tek bitlik farklı türde ve modelde hatalar mikroişlemcinin yazmaç dosyasına (register file) verilerek analizler yapılmıştır. Temel bir yöntem geliştirilerek hatanın tespit edilmesini sağlayan bir devre tasarlanıp mikroişlemci içerisine eklenerek sistemin simülasyon ve sentez sonuçları gösterilmiştir.

Abstract

To implement reliable digital systems, it is necessary to specify the capacity of testability. Particularly it is intensely important to test and detect faults in microprocessors in which be used in major system due to effect on whole system. In this paper, a microprocessor implemented in FPGA and a fault production circuit which is improved by being based fault injection method is entegrated and made a system. The processed faults with different type and model is given into the register file of the microprocessor and doing analysis. By developing a basic method an error detection circuit is designed, attached into the microprocessor and simulation and synthesize results of the system are showed.

1. Giriş

Günümüzde sayısal sistemler Verilog, VHDL gibi Donanım Tanımlama Dilleri (Hardware Description Language-HDL) ile tasarlanmaktadır. Bu sistemlerin gerçek bir donanıma dönüştürülmeden önce test edilebilir olması gerekmektedir. Test edilebilirlik kapasitesi sistemin beklenilen şekilde çalışabilmesi için tasarım değişikliklerinin yapılabilmesini sağlar. Bir hata enjeksiyon yöntemi sistemin beklenilen bölgesine hata vererek test edilebilirlik kapasitesinin belirlenmesini sağlar [1].

Hata türleri devrenin girişinin beklenilenden farklı olmasına ve sistemin başarısızlığa uğramasına sebebiyet verir. Donanımsal hata türleri kalıcı ve geçici olarak sınıflandırılabilir. Bir kalıcı hataya genelde fiziksel bozulmalar, kısa devreler ya da bellekteki bitlerin takılı kalmaları sebep olmaktadır. Geçici hatalar ise kısa bir zaman periyodunda aktif kalırlar. Bilgisayar belleklerinde baskın hatalardır, çevresel etkiler ile oluşmaktadırlar. Hataların etki gösterdikleri şekil hata modelleri ile tanımlanır. Bunlardan en yaygın olan hata modeli takılı kalma (stuck-at-fault) modelidir. Bellek hücresinin veya veri hattının kalıcı veya geçici hata süresince sürekli olarak olarak lojik 1 ya da 0'da kalması ile sonuçlanır [2].

Mikroişlemciler donanım tanımlama dilleri ile tasarlanan sistemlerin önemli bir parçasını oluşturmaktadır ve test edilebilirlik kapasitelerinin belirlenmesi önemlidir. İndirgenmiş Komut Takımı Bilgisayarı (Reduced Instruction Set Computing-RISC) yapısı ile tasarlanan mikroişlemciler genel amaçlı kaydediciler içerir, aritmetik ve mantıksal işlemler bu kaydediciler üzerinden yapılır [3]. İşlemlerin kaydediciler üzerinden yapılması hata geldiğinde tüm sistemi etkilemesi sebebi ile bu çalışmada işlemcinin genel amaçlı kaydedicilerinin sadece tek bir bitine [1]'den temel alınıp geliştirilen bir hata enjeksiyon yöntemi uygulanmıştır.

Sistemin doğru çalışabilmesi için mimarinin önemli bir bölümünü oluşturan yazmaç dosyasının (register file) hatasız çalışması gerekmektedir. Oluşan hataların tespit edilebilmesi için basit bir yaklaşım olarak Hata Tespit Kodları (Error Detection Codes) kullanılabilir [4]. Bu çalışmada ek bir

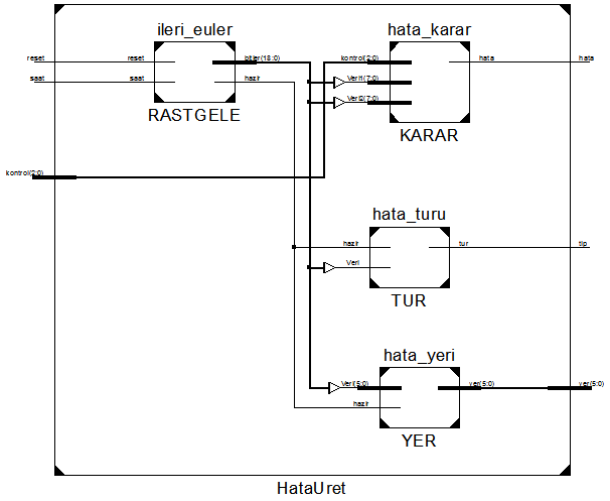
kaydedici kullanılarak geliştirilen hatanın yerini tespit eden temel bir devre tasarlanmıştır.

Çalışmada hazır Natalius adındaki küçük boyutlu 8 bit RISC yapısına dayanan tamamen Verilog dili ile modellenmiş ve Sahada Programlanabilir Kapı Dizileri (Field Programmable Gate Array-FPGA) üzerinde gerçeklenmiş mikroişlemci kullanılmıştır. Natalius 8 bit ALU, 8x8 genel amaçlı kaydedici, 8 bit adres portu, sıfır ve elde bayrağı, 16x11 yığın bellek içermektedir. 2048 adet 16 bitlik komut depolar ve her bir komut 3 saat periyodu ile çalışır [5].

Makalede Kısım 2'de mikroişlemci içerisinde hata üretmek için tasarlanmış modelin algoritmik yapısı ve içerisinde geliştirilmiş bloklar açıklanmıştır. Kısım 3'te bir çarpıcı devre sistemi mikroişlemci üzerine bir çarpıcı algoritması ile gerçeklenmiş ve sistem hata üreteci devresi ile birleştirilerek mikroişlemciye hata verilmesi pratik olarak gözlemlenmiş, Kısım 4'te hata yerinin tespit edilmesi için yedekleme yöntemi geliştirilmiştir. Kısım 5 ise sonuç bölümünü oluşturmaktadır.

2. Hata Üretim Devresi

Hata üretim devresi mikroişlemcinin genel amaçlı kaydedicilerine yönelik hata oluşturmaktadır. Şekil 1'de gösterilen devre 19 bitlik rastgele sayıların üretildiği rastgele sayı üreteci, hata yeri belirleme, hata modeli belirleme ve hata karar bloğundan oluşmaktadır.



Şekil 1: Hata Üretim Devresi.

2.1. Rastgele Sayı Üreteci Devresi

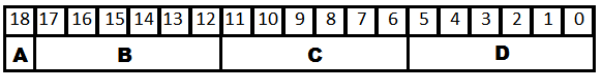
Tasarımda kullanılan kaotik bir davranış sergileyen analog yapıdaki gerçek rastgele sayı üreteci (True Random Number Genarator-TRNG) devresinin sürekli zaman denklemi (1)'de verilmiştir. α çatallaşma ve τ gecikme Ts örnekleme periyodunu ifade eden sistem parametreleridir [6].

$$x'(t) = x(t) + \alpha * f(x(t_k - \tau) \quad t_k \le t \le T_s \qquad (1)$$

Sürekli zamanlı denklem iterasyon ile çözüm yapılmasını sağlayan Euler yöntemi ile ayrıklaştırılıp sayısal donanım olarak modellenmiş devrenin matematiksel ifadesi (2)'de verilmiştir. Denklemde *dt* iterasyon adımını ifade eder küçük olması sistemin sürekli zamanlı sistem davranışına yakın olmasını sağlar. *M* parametresi τ 'nun ayrık zamandaki karşılığıdır [7].

$$x[n+1] = x[n] + dt * \left(x[n] + \alpha * f(x[n-M]) \right) \qquad (2)$$

Doğrusal Geribeslemeli Ötelemeli Kaydediciler (Linear Feedback Shift Register-LFSR) gibi klasik sözde rastgele sayı (Pseudo Random Number Generator-PRNG) üreteci devrelerinin yerine bu modelin kullanılma amacı önceki çalışmalarda yapılan testlerinin sonuçlarına dayanmaktadır. Gerçek rastgele sayı üretecine benzer davranış gösteren bu devre daha kaliteli rastgele sayılar üretmektedir. Bu devrenin ürettiği çıkış bit sayısı arttırılarak sayı uzayı büyültülebilir ve özellikle karmaşık mikroişlemcilerin genel amaçlı kaydedicilerin hata yerinin rastgele seçilmesine olanak sağlar. Üretilen 19 bitlik bir rastgele sayı bloklara ayrılarak oluşturulacak hata oranı, hatanın yeri ve türü belirlenir. Şekil 2'de gösterilen veri bloklarından 1 bitlik A hata modeli, 6'şar bitlik B hata yeri, C ve D ise karar modülüne giriş olarak verilmiştir.



Şekil 2: Rastgele sayının bitlerinin bölümlendirilmesi

2.2. Hata Karar Devresi

Karar devresi dışarıdan kullanıcının girdiği 3 bitlik konrol girişine göre belirlenmektedir. Kontrol girişi "000" seçilirse devre hatasız, "111" seçilirse kalıcı hatalı olarak çalışmaktadır. Bu iki değer arasında kalan değerler seçildiğinde geçici hata oluşmaktadır. Geçici hata oluşma oranları Çizelge 1'de verilmiştir.

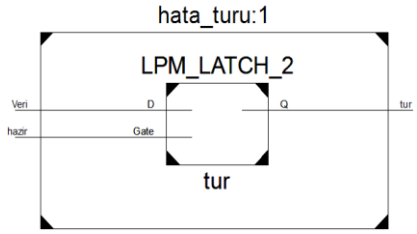
Kontrol girişinin değerine göre rastgele sayılardan gelen verilerin karşılaştırılacak bit sayısı belirlenir. Karşılaştırılan veriler eşit olduğunda hata oluşur. Bit sayılarının artması eşit olma olasılıklarını düşüreceğinden hata oranı orantılı bir şekilde azalır.

Çizelge 1: Karşılaştırılan bitler ve geçici hata oranları

Kontrol Girişi	Hata Oranları (%)
1 (001)	50
2 (010)	25
3 (011)	12.5
4 (100)	6.25
5 (101)	3.125
6 (110)	1.5625

2.3. Hata Modeli Belirleme Devresi

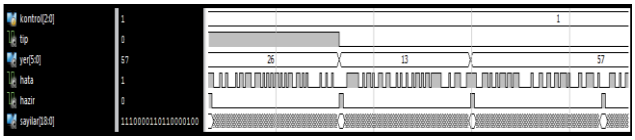
Rastgele sayıların tüm bitlerinin üretildiğini haber veren bayrak geldiğinde gelen verinin değerine göre Şekil 3'de gösterilen devre çıkışının "0"da veya "1"de takılı kaldığı hata modeli belirlenir.



Şekil 3: Hata Modeli Belirleme Devresi.

2.4. Hata Yeri Belirleme Devresi

Hata üretim devresi işlemcinin genel amaçlı kaydedicilerinden herhangi birinin herhangi bir bitine yani tek bir yere gelecek şekilde tasarlanmıştır. Çalışmada ele alınan mikroişlemcinin 8 adet 8 bitlik kaydedicisi, yani hatanın gelebileceği 64 bitlik yer vardır. Rastgele sayı üretecinden gelen 6 bitlik veri, hatanın yerini belirlemektedir.



Şekil 4: Hata üretim devresi test sonucu.

Şekil 4'de hata üretim devresinin simülasyon sonucu gösterilmiştir. Kontrol girişi 1 verilerek %50 oranında hata yaptırılmaktadır. 19 bitlik rastgele sayı hazır olduğunda haber verilmekte, hatanın yeri ve modeli üretilmektedir.

3. Mikroişlemciye Hata Enjekte Etme

Natalius mikroişlemcisinde 30 adet komut kümesi tanımlanmıştır. Bu komutlar kullanılarak bir çarpıcı tasarımı yapılmıştır. Birçok mevcut çarpıcı algoritmasından en anlamlı bit ilk "MSB-first" yöntemi kullanılmıştır.

AxB	A=3	B=6 => 0110
T=0	T=0	
for i=3:0	i=3	T=0 shift B_3=0
T=2T	i=2	T=0 shift B_2=1
if B_i=1		T=3 add A
T=T+A	i=6	T=6 shift B_1=1
end		T=9 add A
		T=18 shift B_0=0

Şekil 5: Çarpıcı algoritması örneği

Şekil 5'de kullanılan çarpıcı algoritmasına örnek olarak verilmiştir. T döngü aracılığıyla üzerinde işlem yapılan ve çarpım sonucunu veren değişkendir. A ve B çarpılacak iki adet 4 bitlik sayıdır. Döngü içerisinde her seferinde T iki katına çıkarılır ve B'nin ilgili biti 1 değerinde ise T sayısına A çarpanı eklenir. Mikroişlemcide her işlem yazmaçlar aracılığı ile yapıldığından ve yazmaçların 8 bitlik olması sebebiyle 8 adet döngü gerçekleşmektedir.

Şekil 6'da ise komut kümesi kullanılarak yazılmış kod hatanın etkilerini gösterecek grafiklerde anlaşılır olması açısından önemlidir. Buna göre r1 ve r2 yazmaçları algoritmadaki A ve B çarpanlarına karşılık gelmekte, port adresine göre değerleri dışarıdan işlemciye giriş olarak verilmektedir. r3 ise T'ye karşılık gelir, sonuç üretildiğinde karşılık gelen port adresine göre işlemci çıkışına verilir. Değeri 1 olan r4 ve başlangıçta 9 olan r6 her seferinde r6'dan r4'ün çıkarılıp karşılaştırılmasıyla 8 adet döngüyü sağlarlar. Değeri 128 olan r5 ise r2'nin en anlamlı bitinin kontrol edilmesini sağlayan sabittir.

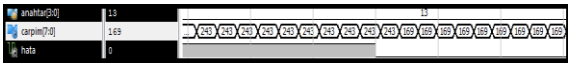
```
forever  csr square
         jmp forever
square   ldm r1,1
         stm r1,1
         ldm r2,2
         csr multsoft
         stm r3,7
         ret
multsoft ldi r3,0
         ldi r4,1
         ldi r6,9
         ldi r5,128
multloop add r3,r3
         ldi r7,0
         oor r7,r2
         oor r7,r5
         cmp r2,r7
         jpz addbit
continue add r2,r2
         sub r6,r4
         cmp r6,r4
         jnz multloop
         ret
addbit   add r3,r1
         jmp continue
```

Şekil 6: Mikroişlemci için yazılan çarpıcı kodu

3.1. Belirli Bir Yazmaca Gelen Hatanın İncelenmesi

Bu bölümde çarpıcı algoritmasında yeri ve modeli sabit bir yazmaca gelen hata incelenip sonuçlar grafikler ile yorumlanmıştır. Analizlerin daha rahat gözlenebilmesi için iki farklı sayı yerine bir sayının karesi alınarak simülasyonlar yapılmıştır. Bu incelemenin yapılma amacı, algoritmada dışarıdan veri alıp veren ve döngüler ile sürekli değeri değişen yazmacın, kalıcı ve geçici hatalar geldiğinde davranışının kestirilip sonucun nasıl etkilendiğini göstermektir.

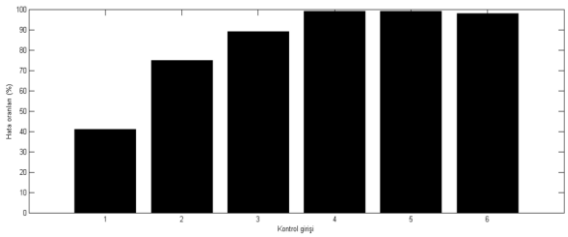
Şekil 7'de r2'nin en anlamlı bitine hata geldiğinde alınan sonuç gösterilmiştir. Çünkü algoritmada en hassas bit bu noktadır. Algoritmada açıklandığı üzere bu noktaya bakarak hesaplanacak sonuca götüren r3 yazmacına, hata modeline göre yanlış ekleme yapılacak ya da hiç ekleme yapılmayacak ve sonuç hatalı çıkacaktır.



Şekil 7: r2 yazmacına gelen hata ile ve hatasız çarpım sonucu

Şekil 8'de kontrol girişine bağlı olarak sonuçların doğruluk oranları gösterilmiştir. Her bir kontrol girişine göre 100'er adet farklı rastgele sayıların üretilmesiyle veriler alınarak sonucun

doğruluk oranları gösterilmiştir. Verilen hata oranına yakın doğruluk yüzdelerinin çıktığı görülebilmektedir.



Şekil 8: Konrol girişine bağlı sonuçların doğruluk oranları.

Çizelge 2'de ise kontrol girişi 1 iken r2 yazmacının en anlamlı bitinde yapılan koşullu hata olasılıkları verilmektedir. Bu olasılık hata oranı ile sayıların içerdikleri 1 veya 0 sayısının toplam bit sayısına bölünmesi ile elde edilir. Örneğin 15 sayısının ikili gösterimi "1111"dir ve 4 adet 1 içerir. Yazmaç 8 bitlik olduğundan verinin %50'si 1 içermektedir. Hata oranı ise kontrol girişinin 1 olmasından dolayı %50'dir. Yani bu veri için yazmaç ikisinin çarpımı sonucu çıkan %25'lik oran ile hata yapar.

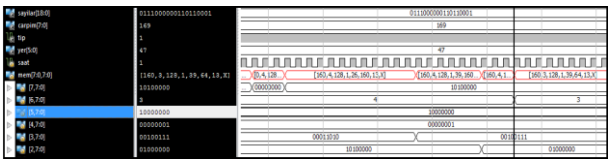
Çizelge 2: %50 oranında geçici hatada yazmacın hata oranı

4 bitlik veriler	15	14,13, 11,7	12,10, 9,6,5	8,4, 2,1	0
1/0 adedi	4/4	3/5	2/6	1/7	0/8
1/0 takılı kalma oranları (%)	25/ 25	31.25/ 18.75	37.5/ 12.5	43.75/ 6.25	50/ 0

3.2.Rastgele Bir Yazmaca Gelen Hatanın İncelenmesi

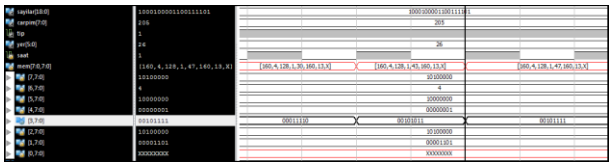
r0'dan r7'ye kadar olan 8 bitlik yazmaç dosyasına kullanıcının belirlediği kontrol girişine göre hata üretim devresinde rastgele belirlenen modelde ve yerde hata oluşur.

Bazı hatalar bulunduğu yere ve modele göre maskelenebilir. Böyle bir durumda sistem normal olarak çalışmasına devam eder. Şekil 9'da r5'in en anlamlı bitine "1"de takılı kalma hatası gelmiş ve çarpım sonucu 169 olarak doğru çıkmış yani hata maskelenmiştir.



Şekil 9: r5 kaydedicisine gelen hatanın davranışsal simülasyonu

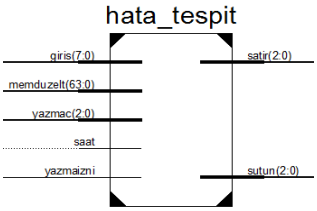
Şekil 10'da ise r3'ün 3. bitine "1"de takılı kalma hatası gelmiştir. Değeri 0 olan bit hatanın gelmesiyle birlikte 1'e dönüşmüş ve çarpım değeri 205 olarak hesaplanmış yani çıkış yanlış sonuçlanmıştır.



Şekil 10: r3 kaydedicisine gelen hatanın davranışsal simülasyonu

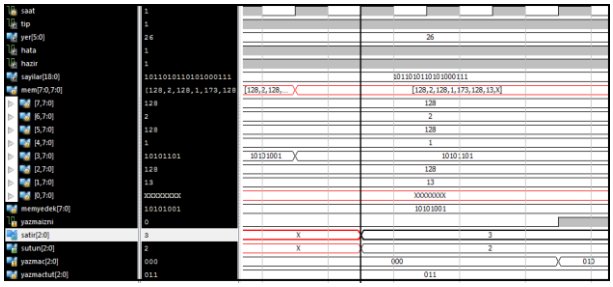
4. Mikroişlemcideki Hatanın Yedekleme Yöntemi ile Tespit Edilmesi

Yazmaçlara gelen rastgele bir hatanın yerinin bulunabilmesi için basit bir hata tespit devresi tasarlanıp mikroişlemci içerisine eklenmiştir. Yazmaç dosyasına giden 8 bitlik girişler aynı şekilde Şekil 11'de gösterilen devre içerisine de gelmekte, yazma izni verildiğinde yedek bir kaydediciye alınmaktadır. Bir dahaki yazma iznine kadar yedek kaydedicide tutulan bu veriler yazmaç dosyasındaki tüm yazmaçların memduzelt adındaki hatta atandığı veriler ile 8'er bit olarak karşılaştırılmakta eşit olmadığı yerde hatanın hangi yazmaçta olduğunu satir, hangi bitinde olduğunu sutun adlı çıkışta göstermektedir.



Şekil 11: Hata tespit devresi

Şekil 12'de simülasyon sonuçları gösterilmiştir. Hatanın yeri 26 olarak üretilmiştir ve bu da r3 yazmacının en anlamsız 3. bitini ifade etmektedir. Hata yeri satir ve sutun çıkışlarında tespit edilmiştir.



Şekil 12: Hata tespit devresi simülasyon sonuçları

5. Sonuçlar

Çalışmada Verilog ile gerçeklenen bir mikroişlemciye çarpıcı kodu yazılıp tek bir bölgesine hata enjekte edilerek test edilebilirlik kapasitesi ölçülmüştür. Algoritmasından baz alınan hata enjekte yönteminden yola çıkılarak geliştirilen hata üretim devresinin istenilen yerde ve modelde hata üretebildiği gösterilmiş ve bu hatanın tespit edilebilmesi sağlanmıştır. Analizler mikroişlemcideki tüm aritmetik ve lojik işlemlerin yapıldığı en hassas bölge olan kaydedici dosyasında yapılmıştır. Devre içerisinde analog yapıdan Euler yöntemi ile ayrıklaştırılarak sayısal olarak tasarlanan rastgele sayı üretecinden çıkan bitler bölümlendirilerek hata yerini ve modelini, ayrıca geçici hatalar için de kontrol girişinin değerine göre hata oranını belirlemektedir. Öncelikle algoritmadaki döngülerde sürekli değeri değişen yazmacın en anlamlı bitine hata enjekte edilerek gelen veriye göre yazmacın hata oranı belirlenmiştir. Ardından herhangi bir yazmaca gelen hatanın geldiği yere ve modele göre yanlış sonuca neden olduğu ya da maskelenip sonucu etkilemediği gösterilmiştir. Çalışmanın son aşamasında ise yedek bir kaydedici mikroişlemciye eklenerek hata tespit mekanizması oluşturulmuştur. Gelecek çalışmalarda gelişmiş mikroişlemcilerin herhangi bir bölgesinde ve çoklu oluşabilecek hataları incelemek için temel oluşturulmuştur.

6. Kaynaklar

[1] Lala, P., "Transient and Permanent Fault Injection in VHDL Description of Digital Circuits", Scientific Research, Cilt No., 192-199, 2012.

[2] Dubrova, E., Fault Tolerant Design, Springer, New York, 2013.

[3] Nurmi, J., *Processor Design*, Springer, New York, 2013.

[4] Sorin, D.H., Fault Tolerant Computer Architecture, Morgan and Claypol Publishers, Madison, 2009.

[5] Natalius 8 bit RISC [Online]. Available: http://opencores.org/project,natalius_8bit_risc

[6] Yeniçeri, R., Ustaoğlu, B., and Yalçın M.E "Throughput Enhancement for a New Time-delay Sampled-data Based True Random Bit Generator", *European Conference on Circuit Theory and Design*, 2013, 1-4.

[7] Ustaoğlu B. *Gerçek Rastgele Sayı Üreteci Tasarımı, Testleri ve Gerçeklenmesi,* Bitirme Tezi, İstanbul Teknik Üniversitesi Elektrik Elektronik Fakültesi, 2013.

Ek A

Sentez Sonuçları

Devrelerin Xilinx ISE 13.2 versiyonunda ve Spartan-3e500 kartında yapılan sentez sonuçları verilmektedir. Sentez sonuçları devrenin karta yerleşiminden önce tahmini kapladığı alan ve kullandığı blokları gösterir. Şekil 13'de hata üretim devresi karmaşık sayısal sistemler içerisinde kullanılabilecek şekilde az yer kapladığı görülebilmektedir. Şekil 14'te mikroişlemciye çarpıcı kodu yazılmış ile hata üretim devresinin birleştirilmiş tüm sistemin, Şekil 15'te mikroişlemciye hata tespit mekanizması katılmış haldeki sentez sonuçları verilmiştir. Aradaki farkın az olması hata tespit devresinin sistemde az miktarda alan kapladığı sonucunu gösterir.

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	90	4656	1%
Number of Slice Flip Flops	79	9312	0%
Number of 4 input LUTs	162	9312	1%
Number of bonded IOBs	13	232	5%
Number of GCLKs	2	24	8%

Şekil 13: Hata üretim devresi sentez sonucu

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	354	4656	7%
Number of Slice Flip Flops	198	9312	2%
Number of 4 input LUTs	650	9312	6%
Number of bonded IOBs	13	232	5%
Number of BRAMs	2	20	10%
Number of GCLKs	2	24	8%

Şekil 14: Mikroişlemci ve hata üretim devresini içinde barındıran çarpıcı sisteminin sentez sonucu

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	403	4656	8%
Number of Slice Flip Flops	216	9312	2%
Number of 4 input LUTs	745	9312	8%
Number of bonded IOBs	23	232	9%
Number of BRAMs	2	20	10%
Number of GCLKs	2	24	8%

Şekil 15: Mikroişlemciye hata tespit bloğunun eklenmiş çarpıcı sisteminin sentez sonucu