Çukurova Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 29(1), 157-163 ss., Haziran 2014
Çukurova University Journal of the Faculty of Engineering and Architecture, 29(1), pp. 157-163 , June 2014

FPGA'lar için Ardışık Büyük Tam Sayı Kare Alıcılar

Ali ŞENTÜRK [*1], Mustafa GÖK [2],

[1]*Ç.Ü., Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Adana*
[2]*Ç.Ü., Mühendislik-Mimarlık Fakültesi, Elektrik Elektronik Mühendisliği Bölümü, Adana*

Özet

Bu çalışmada FPGA'lar için ardışık büyük tam sayı kare alıcı tasarımı gösterilmiştir. Tasarım boru hatlı yapıdadır ve çoğu modern FPGA'larda bulunan gömülü çarpıcıları kullanmaktadır. Tasarımda, geliştirdiğimiz ayrıştırma yöntemi ile gömülü çarpıcıların daha verimli kullanılması sağlanmıştır. Tasarım yarım veya tam büyüklükte sonuç üretebilmektedir. Sentez sonuçları, Ardışık Büyük Tam Sayı Kare Alıcı uygulamalarının, önceki çalışmalarda gösterilen ardışık büyük çarpıcılarla yaklaşık aynı kaynak tüketimine sahip olmasına rağmen, %62'ye varan yüksek performans kazançları sergilediğini göstermiştir.

Anahtar Kelimeler: FPGA, Boru hatlı tasarım, Ardışık büyük kare alıcı

Large Sequential Integer Squarers for FPGAs

Abstract

This paper presents sequential large integer squarer designs for FPGAs. The design is pipelined and uses embedded multiplier blocks which exist in most modern FPGAs. More efficient use of embedded multipliers is provided with the decomposition method which we developed in the design. The design can generate a full size or a single size product. The syntheses results show that compared to sequential large multipliers presented in the previous researches, Sequential Large Integer Squarers consume almost same resources and have up to 62% higher performance.

Anahtar Kelimeler: FPGA, Pipelined design, Sequential large squarer

[*] Yazışmaların yapılacağı yazar: Ali ŞENTÜRK, *Mühendislik Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Adana. asenturk@cu.edu.tr*

1. GİRİŞ

Büyük sayılar kriptografik ve bilimsel çalışmalarda sıklıkla kullanılmaktadır[1-3]. Çarpmanın özel bir formu olan kare alma işlemi bu uygulamalarda kullanılan en yaygın aritmetik işlemlerden biridir. Genel amaçlı bilgisayarlarda büyük sayıları işleyen aritmetik donanımlar bulunmaz. Bu donanımsal eksikliği telafi etmek için, özel kütüphaneler kullanılmaktadır [4]. Genel olarak büyük kare alma rutinlerinin büyük sayıları standart kelime boyutlarında parçalara ayrıştırmasıyla, küçük çarpımlar gerçekleştirilmektedir. Parça çarpma işlemleri sonucu elde edilen çarpımlar hizalanıp toplanmaktadır. Temel ayrıştırma yönteminden daha hızlı algoritmalar da [5] mevcuttur. Fakat bunlar da işlemcide bulunan standart büyüklükteki çarpıcılar kullanılarak gerçekleştirilmektedir. Eğer büyük sayılarla çok fazla aritmetik işlem yapılacaksa, hızlı algoritmalarla olsa bile, yazılımsal yaklaşımlar zaman kaybına neden olabilir. Bundan dolayı donanımsal kaynakları verimli kullanan, yüksek hız sağlayan aritmetik tasarımlara ihtiyaç vardır.

Son yapılan büyük çarpıcı - kare alıcı ile ilgili çalışmalar, hızlı tasarım ve esneklik avantajlarından dolayı FPGA uygulamalarında yoğunlaşmıştır [6-9]. Tasarımlarda ayrıştırma yöntemi ile böl-fethet veya Karatsuba-Ofman çarpım algoritmaları kullanılmıştır. Çalışmalar arasında ardışık tasarımlar bulunsa da çalışmaların çoğu kombinasyonel tasarımlarda yoğunlaşmıştır. Daha önce yapılan çalışmalar kısaca aşağıda özetlenmiştir.

Çalışma [6]'da ayrıştırma metodunu kullanan kombinasyonel büyük çarpıcı ve kare alıcı tasarımları gösterilmiştir. Tasarımlarda parça çarpımlarından üretilen sonuçların toplanması için toplayıcı ağaçları kullanılmaktadır. Toplayıcı ağacı kısmi çarpımların toplanmasını hızlandırmaktadır. Çalışma [6]'da sentez sonuçlarının 20 bit'den 85 bite kadar olduğu ve Spartan-3 FPGA'lere aktarıldığı raporlanmıştır.

Çalışma [7]'de ayrıştırma metodunu kullanan bir başka kombinasyonel çarpıcı tasarımı gösterilmiştir. Ayrıştırma işlemi, aritmetik dilimlerin yapısına ve Virtex-4 FPGA'leri tarafından sağlanan hızlı elde zincirlerine göre yapılmıştır. 16 bitden 221 bite kadar büyüyen tasarımlar FPGA'ler üzerine aktarılmıştır. FPGA üreticisinin sağlamış olduğu araçlarla sentezlenen çarpıcılarla karşılaştırıldığında, yapılan uygulamalar daha yüksek güç ve donanım kullanmaktadır ama üç kata kadar daha hızlıdır.
Çalışma [8]'de verimli bir kare alma metodu önerilmiş ve bu metodun, döşeme metodundan %21'e kadar daha az LUT'a gereksinim duyduğu raporlanmıştır. Çalışma [8] Virtex 6 FPGA'ler hedeflenerek sentezlenmiştir. Boru hatlı yapıdaki tasarımda uygulamalar 64 bite kadar çıkmaktadır. Uygulamanın maksimum frekansı 442 MHz'dir.

Çalışma [9]'da FPGA'ler için üç tane ardışık büyük çarpıcı tasarımı önerilmiştir. Çalışma, ayrıştırma metodunu kullanmakta ve üç tasarımın hız-alan karşılaştırmalarını göstermektedir. 256 bitden 2048 bite kadar uygulamalar sentezlenmiş ve Virtex 5 FPGA'lere aktarılmıştır. Sentezlenen 256 bit çarpıcının, FPGA üreticisinin aracı ile sentezlenen 256 bitlik çarpıcıya göre yaklaşık 3 kat yavaş olduğu ve 15 kat daha az kaynak kullandığı belirtilmiştir.

Literatür incelemeleri, daha önceki büyük kare alıcı, çarpıcı çalışmaların daha çok, donanımsal kaynakların rahatça tüketildiği, performansa yönelik olarak, kombinasyonel tasarımlarla yapıldığını göstermektedir. Şu anda, tüm donanımsal kaynaklar kullanıldığında bile Virtex 5 FPGA'ler üzerinde sentezlenebilecek en büyük kombinasyonel çarpıcının boyutu 256 bittir. Fakat pratikte tüm FPGA kaynakları sadece bir çarpıcı için ayrılamaz. Bir başka husus da daha önceki çalışmalarda gösterilen performans geliştirmeleri, hızlı elde zincirleri ve özellikle döşeme optimizasyonu kullanan tasarımlardaki gömülü çarpıcı boyutları gibi platforma özel fonksiyonlardan oluşmaktadır. Genelde aynı FPGA ailesinin üyelerinde bile donanımsal değişiklikler mevcuttur. Bu durumlar göz önüne alındığında ardışık tasarımlar, kombinasyonel tasarımlara iyi bir alternatif oluşturmaktadır. Ardışık tasarımlar göreceli olarak az kaynak

gerektirmektedir. Saklama için yeterli kaynak olduğu sürece herhangi büyüklükte çarpım ardışık çarpıcılar kullanılarak yapılabilir. Ayrıca bu metotlarla yapılan kaynak kullanımları çoğu FPGA aileleri için neredeyse aynıdır. Doğal olarak ardışık tasarımlar, kombinasyonel tasarımlara göre daha yavaştır. Fakat boru hatlı tasarımlar ve modern FPGA'lerde sağlanan yüksek hızlı aritmetik dilimlerle ardışık tasarımların performansları geliştirilebilir. Bu çalışmada, belirtilen tasarım yaklaşımları kullanılarak ardışık kare alıcı tasarımı yapılmıştır. Kare alıcı tasarımı ayrıştırma metodu üzerine bina edilmiştir. Tasarımın 512, 1024 ve 2048 bit uygulamaları Virtex 5 FPGA'ler üzerine aktarılmıştır. Makalenin geri kalanı şu şekilde organize edilmiştir: Bölüm 2'de büyük kare alıcı tasarımının algoritması ve tasarımı gösterilmiştir, Bölüm 3'de tasarımların farklı boyutlardaki sentezleri için gerekli olan donanımsal kaynaklar ve hız sonuçları raporlanmıştır, Bölüm 4'te çalışmanın sonucu gösterilmiştir.

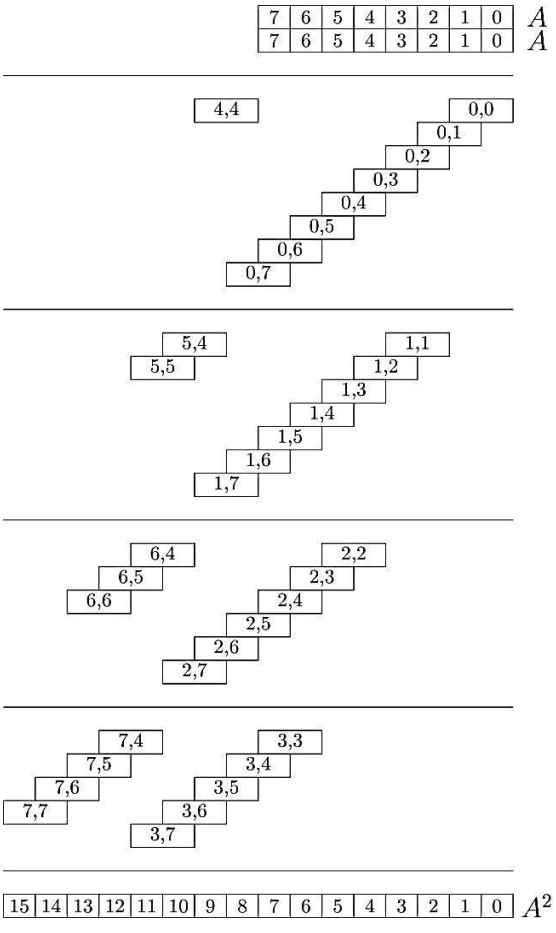
2. ARDIŞIK BÜYÜK KARE ALMA

Kare alma işlemi çarpılan ve çarpan sayının aynı olduğu çarpmanın özel bir formudur. Kare alma işlemi için w bit uzunluğunda A sayısı n bit parçalara ayrıştırılabilir. Parçalar Eşitlik 1'de gösterildiği gibi çarpılıp toplanır.

$$A^2 = \sum_{i=0}^{p-2} \sum_{j=i+1}^{p-1} 2 \cdot A_i \cdot A_j \cdot 2^{n\cdot(i+j)} + \sum_{i=0}^{p-1} A_i^2 \cdot 2^{2\cdot n\cdot i} \qquad (1)$$

Eşitlik 1'de $A_i \cdot B_j = A_j \cdot B_i$ olduğu için, A^2 nin hesaplanması için, $p = \lceil w/n \rceil$ olmak üzere, $n \cdot (n+1)/2$ çarpıma ihtiyaç vardır. Hesaplamada p tane n bit boyutlu küçük çarpıcı aynı anda çarpma işlemleri için kullanılırsa, Eşitlik 1'in ardışık uygulamasındaki çarpma işlemleri p iterasyonda tamamlanır. Bununla birlikte belirtilen şekilde yapılacak olan doğrudan uygulamada, hesaplamaya etkin olarak katkıda bulunan küçük çarpıcı sayısı her iterasyonda bir azalmakta ve küçük çarpıcılar kare alma işleminin bütün iterasyonlarında verimli bir şekilde kullanılamamaktadır. Küçük çarpıcıların her iterasyonda etkin olarak kullanılması ve kare alma işleminin daha hızlı gerçekleştirilebilmesi amacıyla ayrıştırma yönteminde düzenlemeler yaptık. Bazı parçaların çarpım sırasını değiştirerek, boşta kalan çarpıcıları sırası değiştirilmiş olan parçaların çarpımında kullandık. Alt çarpımlar birbirinden bağımsız olduğu için, sıra değiştirme işleminin sonuca herhangi bir etkisi olmadı. Sırası değiştirilerek yapılan alt çarpmalar da dahil, tüm çarpma işlemlerinin sonuçları hizalanıp toplanarak çarpım sonucu üretildi.

Parçaların çarpımı örneği Şekil 1'de gösterilmiştir. Bu örnekte büyük sayı A, 8 parçaya ayrıldı. Parça çarpımları için 9 küçük çarpıcı devresi kullanıldı.



Şekil 1. Büyük kare alma için alt çarpımların hizalanması

Kutu içerisindeki gösterilen sayı çiftleri parça çarpımlarını göstermektedir. Örneğin, 0,1 A_0 ve A_1 çarpımını gösterir. Birinci iterasyonda, sekiz küçük çarpıcı kullanılarak $A_7A_6 \dots A_1A_0$ parçaları A_0 ile çarpılmaktadır. Dokuzuncu çarpıcı $A_4.A_4$ çarpımı için kullanılır. Tüm iterasyonlarda birinci ve dokuzuncu haricindeki tüm çarpıcıların sonuçları bir bit sola kaydırılır. İkinci iterasyonda $A_7A_6 \dots A_1$ parçaları A_1 ile çarpılır. $A_0.A_1$ çarpımı birinci iterasyonda gerçekleştirildiği ve 1 bit sola kaydırma ile çarpımın iki katı elde edildiği için $A_1.A_0$ çarpımı gerçekleştirilmez. Bu iterasyonda sekizinci ve dokuzuncu çarpıcılar sırası değiştirilmiş $A_4 \cdot A_5$ ve $A_5 \cdot A_5$ çarpımları için kullanılır. Bu kalıp son iterasyona kadar kullanılır. Şekil 1'de gösterildiği gibi hizalanmış parçaların toplamıyla da işlem sonucu elde edilir. Sırası düzenlemiş kare alma işlemi için genel formül Eşitlik 2'de verilmiştir.

$$A^2=\sum_{i=0}^{p/2-1}\left\{\begin{array}{c}\sum_{j=i+1}^{p-1}\left[A_i\cdot A_j\cdot 2^{n\,(i+j)+1}\right]+A_i^2+\\ \sum_{j=0}^{i-1}\left[A_{p/2+i}\cdot A_{p/2+j}\cdot 2^{n\,(p+i+j)+1}\right]+\\ A_{p/2+i}^2\end{array}\right\} \quad (2)$$

Büyük tam sayı kare alıcısı için gerekli olan adımlar Algoritma 1'de gösterilmiştir. Algoritma iki bölümden oluşmaktadır. Birinci bölüm sonucun düşük değerlikli yarısını hesaplamaktadır. Bu bölümde bulunan dış döngü zamana bağlı iterasyonları, iç döngüler ise üniteler arasındaki bağlantıları göstermektedir. Dış döngünün her iterasyonunda sonucun $2n$ bitlik kısmı üretilmektedir. Algoritmanın birinci bölümü sonucun w bit uzunluğunda düşük değerlikli yarısı üretilene kadar çalıştırılmaktadır. Algoritmanın ikinci bölümü ise yüksek değerlikli yarının hesaplanması için kullanılmaktadır. Birinci bölümde elde edilen toplam (S) ve elde (C) değerleri ikinci bölümde hizalanıp toplanmaktadır. Algoritmada alt indislerdeki köşeli parantezli gösterim bit aralığını ifade etmektedir. 29. satırdaki süslü parantezle gösterilen alt indis ise $n-2$ tane 0 ile oluşturulmuş diziyi ifade etmektedir. '&' operatörü birleştirme için kullanılmıştır.

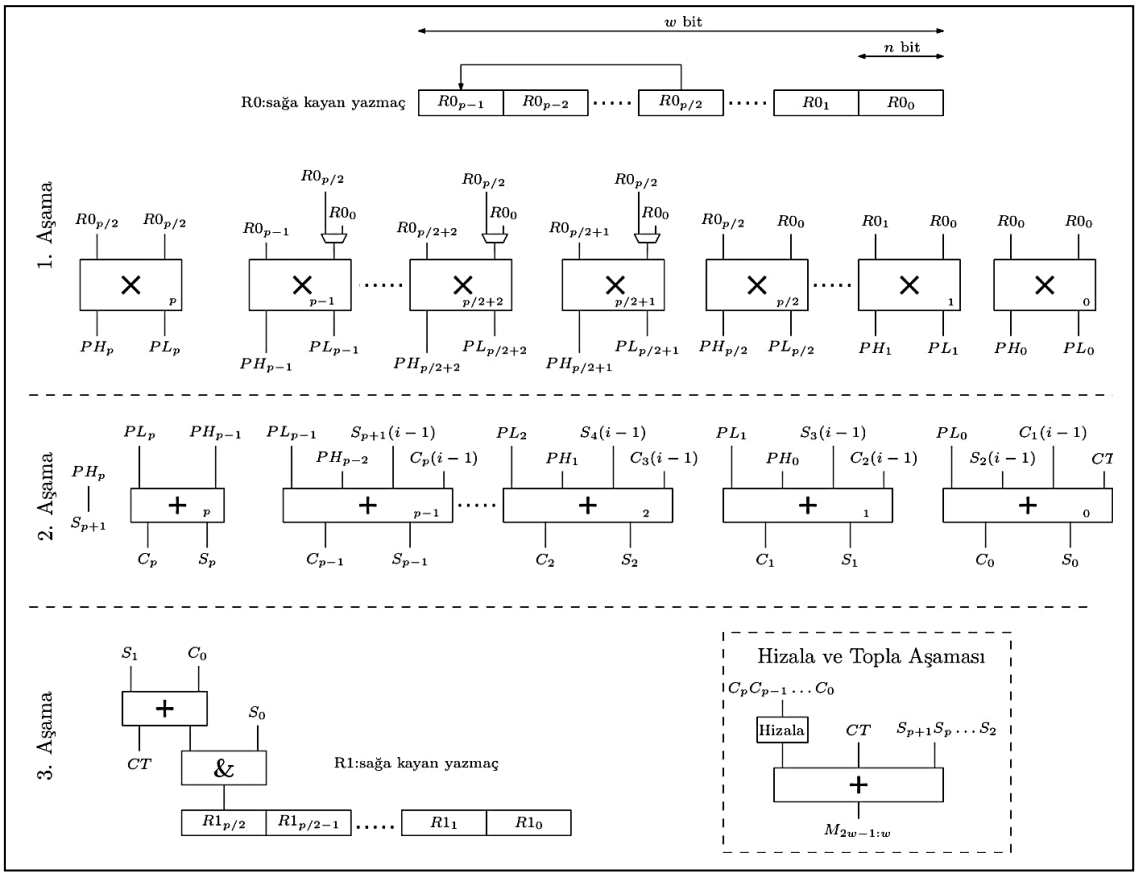
Algoritma 1 Büyük Tam Sayı Kare Alıcı Uygulama Algoritması
$S = 0, C = 0, SH = 0, CT = 0$

Birinci Bölüm

1. **for** $i = 0$ **to** $(p/2-1)$ **do**
2. **for** $j = i$ **to** $p-1$ **do**
3. $P(i,j) = A(i) \cdot A(j)$
4. **if** $j > i$ **then**
5. $P(i,j) = 2 \cdot P(i,j)$
6. **end if**
7. $PH(i,j) = P(i,j)_{[2n:n]}$
8. $PL(i,j) = P(i,j)_{[n-1:0]}$
9. $S(i+j) = PL(i,j) + PH(i,j-1) + C(i+j-1) + S(i+j)_{[n-1:0]} + CT$
10. $C(i,j) = S(i,j)_{[n+2:n]}$
11. $CT = 0$
12. **end for**
13. $M_{[2n(i+1):2n.i]} = S(i,i+1) + C(i,i))\&S(i,i)$
14. $CT = M_{[2n(i+1)]}$
15. **for** $k = 0$ **to** i **do**
16. $P(p/2+i, p/2+k) = A(p/2+i) \cdot A(p/2+k)$
17. **if** $k < i$ **then**
18. $P(p/2+i, p/2+k) = 2 \cdot P(p/2+i, p/2+k)$
19. **end if**
20. $PH(p/2+i, p/2+k) = P(p/2+i, p/2+k)_{[2n:n]}$
21. $PL(p/2+i, p/2+k) = P(p/2+i, p/2+k)_{[n-1:0]}$
22. $S(p+i+k) = PL(p/2+i, p/2+k) + PH(p/2+i, p/2+k-1) + S(p+i+k) + C(p+i+k-1) + PH(i,p)$
23. $C(p+i+k) = S(p+i+k)_{[n+1:n]}$
24. $PH(i,p) = 0$
25. **end for**
26. **end for**

İkinci Bölüm

27. **for** $i = 0$ **to** $p-1$ **do**
28. $SN_{[(i+1)n:i.n]} = S(i)$
29. $CN_{[(i+1)n:i.n]} = 0_{\{n-2\}}\&C(i)$
30. **end for**
31. $M_{2w-1:w} = SN + CN + CT$



Şekil 2. Ardışık büyük tam sayı kare alıcı boru hattı aşamaları

3. ARDIŞIK BÜYÜK TAM SAYI KARE ALICI UYGULAMASİ

Önerilen ardışık büyük sayı kare alıcının blok diyagramı Şekil 2'de gösterilmiştir. Tasarım üç boru hattı aşaması ve boru hattından bağımsız bir son aşamadan oluşmaktadır. Aşamalarda bulunan üniteler ve fonksiyonları aşağıda anlatılmıştır:

Birinci Aşama: Karesi alınacak büyük sayı A, w bit uzunluğundaki, kayan yazmaç $R0$'da saklanmaktadır. $R0$ her çevrimde n bit sağa kayar ve $(p/2)$'inci n bitlik dizisini $(R0_{p/2})$, en yüksek değerlikli n bitlik kısma $(R0_{p-1})$ kopyalar. Bu işlem, bir önceki bölümde anlatılan, sırası değiştirilmiş çarpımlar için gerekmektedir. Birinci Aşamadaki çarpıcılar 0'dan p'ye kadar numaralandırılmıştır. Çarpıcılara $R0$'dan yapılan bağlantılar şu şekildedir:

- $0 \leq i \leq p/2$ olmak üzere i. çarpıcının sağ girişlerine $R0_0$, sol girişlerine $R0_i$ bağlanır

.

- $p/2+1 \leq i \leq p-1$ olmak üzere i. çarpıcının sağ girişine $R0_0$ veya $R0_{p/2}$'yi seçen veri seçici bağlanır. Sol girişlerine ise $R0_i$ bağlanır.

- Çarpıcı p'nin her iki girişi da $R0_{p/2}$'ye bağlanır.

İlk ve son çarpıcı hariç diğer çarpıcıdan elde edilen çarpımlar bir önceki bölümde anlatıldığı gibi, iki ile çarpmak için 1 bit sola kaydırılır. Sonrasında tüm çarpımlar 2 parçaya ayrılır. Çarpıcı i'nin düşük değerlikli n bitlik kısmı PL_i ve geri kalan

yüksek değerlikli kısmı PH_i ile gösterilmiştir.

İkinci Aşama: Bu aşamada (p+1) adet toplayıcı aynı değerlikli önceki aşamada elde edilen PH ve PL değerleri ile önceki çevrimde elde edilen toplam değerleri, S(i-1), ve elde değerleri, C(i-1)'leri toplar. Toplayıcılar (n+2) bit sonuç üretirler. Toplayıcı i'nin ürettiği sonucun n bitlik düşük değerlikli kısmını S_i, geri kalan 2 bitlik yüksek değerlikli bitlerini C_i temsil etmektedir. $1 \leq i \leq p-1$ olmak üzere, Toplayıcı i'nin girişleri PH_{i-1}, PL_i, bir önceki iterasyonda üretilen Toplayıcı $(i+2)$'nin S sonucu, $S_{i+2}(i-1)$, ve bir önceki iterasyonda üretilen Toplayıcı $(i+1)$'in C sonucu, $C_{i+1}(i-1)$'dir. Toplayıcı 0 ve Toplayıcı p'nin girişleri diğer toplayıcılara göre farklıdır. Toplayıcı 0'ın girişleri PL_0, $S_2(i-1)$, $C_1(i-1)$ ve 3. aşamadaki toplayıcından elde edilen CT'dir. Toplayıcı p'nin girişleri ise PH_{p-1} ve PL_p'dir.

Üçüncü Aşama: Bu aşamada n bitlik toplayıcı S_1 ve C_0 değerlerini toplar. Toplam S_0 ile birleştirilir. Elde edilen $2n$ bit boyutundaki değer $R1$ yazmacının $2n$ bitlik en yüksek değerlikli kısmında saklanır. $R1$ her iterasyonda $2n$ bit sağa kaydırılır.

Hizala ve Topla Aşaması: Bu aşamada çarpımın yüksek değerlikli yarısı hesaplanır. Boru hattı aşamalarındaki iterasyonlar bittiğinde, yazmaçlara yazma izni verilmez. Boru hattı yazmaçlarında saklanmış olan toplam ve elde değerleri elde verilerinin önlerine $(n-3)$'er tane sıfır konularak hizalanır. Şekil 2'de kesikli çizgili kutuda gösterildiği gibi, hizalanan S, C ve CT eklenerek çarpımın yüksek değerlikli yarısı elde edilir.

4. BULGULAR VE TARTIŞMA

Bu kısımda Ardışık Büyük Tam Sayı Kare Alıcı sentez sonuçları ve önceki çalışmalarla karşılaştırmaları gösterilmektedir. Önerilen tasarımların VHDL modelleri yazılmış ve modeller Xilinx ISE araçları kullanılarak sentezlenmiştir. Çizelge 1'de 512, 1024, 2048 bit Ardışık Büyük Tam Sayı Kare Alıcı tasarımlarının sentez sonuçları gösterilmiştir. Virtex 5 xc5vfx100t hedef cihaz olarak seçilmiş ve sentezler hız optimizasyonu ile gerçekleştirilmiştir. Çizelge 1'in sütunlarında sayıların bit uzunlukları, sonucun elde edilmesi için gerekli olan çevrim sayısı ve toplam süre, yazmaçların, LUT ve DSP dilimlerinin kullanım miktarı ve yüzdeleri gösterilmiştir. Çizelge 1'de gösterilen bütün sentezlerde periyotlar 4,796 nano saniye olarak elde edilmiştir.

Toplam gecikme $(p/2+3+CPA\ Çevrimleri) \cdot Periyot$ formülü kullanılarak hesaplanmıştır. Çizelgede verilen sonuçlarda görüldüğü gibi bit uzunluğunun iki katına çıkması toplam gecikmeyi 1,88 katına kadar çıkarmaktadır. Aynı zamanda kullanılan DSP dilimlerini de yaklaşık 2 katına çıkmaktadır.

Çizelge 1. Ardışık büyük kare alıcı sentez sonuçları

Boyut	Çevrim	Gecikme (ns)	FF	LUT	DSP
512	23	95,124	2900	2480	33
1024	40	169,064	5555	4805	63
2048	75	317,944	10865	9455	123

Bölüm 1'de belirtildiği gibi yapmış olduğumuz tasarım daha önce yapılan çalışmalarla hem hedef cihaz hem de tasarım boyutu olarak farklıdır. Aynı zamanda literatürde büyük kare alma ile ilgili çok fazla çalışma bulamadık. Bundan dolayı Çizelge 2'de aynı tasarım stratejisine sahip Çalışma [9] ile Ardışık Büyük Tam Sayı Kare Alıcı karşılaştırılması yapılmıştır.

Ardışık çarpıcı ile yapılacak olan kare alma işlemine göre, birbirine yakın donanım kullanımı ve 1024 bit sayılar için yaklaşık %50, 2048 bit sayılar için yaklaşık %62 performans artışı sağlanmıştır.

Çizelge 2. Ardışık büyük kare alıcı ve ardışık büyük çarpıcı karşılaştırması

Boyut	Çalışma	Gecikme	Donanım Yüzdeleri
1024	Önerilen Tasarım	169	%9 FF, %8 LUT, %25 DSP
	Çalışma [9]	265	%8 FF, %4 LUT, %23 DSP
2048	Önerilen Tasarım	317	%17 FF, %15 LUT, %48 DSP
	Çalışma [9]	515	%16 FF, %9 LUT, %47 DSP

5. SONUÇLAR

Bu çalışmada FPGA üzerine aktarılan ardışık büyük tam sayı kare alıcı tasarımı anlatılmıştır. Tasarım modern FPGA'ler üzerinde bulunan özel çarpıcı ünitelerini etkin olarak kullanmaktadır. 2048 bit'e kadar kadar sayı uzunluğunu destekleyen tasarımlar mevcut donanımsal kaynakları aşmadan FPGA'ler üzerine sentezlenebilmiştir. Teklif edilen tasarımların hızları kombinasyonel tasarımlara yaklaşan sonuçlar vermiştir. Bununla birlikte, FPGA'ler üzerine sentezlenen en büyük kombinasyonel çalışmamızda hedef cihaz olarak seçtiğimiz FPGA için yaklaşık 256 bit'tir. Bu da büyük sayıların yalnızca ardışık devrelerle sentezlenebileceği anlamına gelmektedir. Sentezlenen ardışık kare alıcı tasarımı, kare alma için gerekli olan iterasyon sayısını düşüren, önerdiğimiz ayrıştırma yöntemini kullanmaktadır. Bu yöntemle aynı yapıdaki bir çarpıcı devresiyle yaklaşık eşit kaynakla, çarpıcı devresinden %62'ye kasar daha hızlı kare alabilen bir ünite tasarlanmıştır.

6. KAYNAKLAR

1. N. Koblitz, 1987. Elliptic Curve Cryptosystems, Mathematics of Computation, 48, s. 203-209.

2. R. L. Rivest, A. Shamir, and L. Adleman, 1978. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications of the ACM, 21, s. 120-126.

3. V. S. Miller, 1986. Use of Elliptic Curves in Cryptography, in Advances in Cryptology—CRYPTO'85 Proceedings, s. 417-426.

4. GMP. 2014. The GNU Multiple Precision Arithmetic Library, Version 6.0. https://gmplib.org/

5. T.-J. Chang, C.-L. Wu, D.-C. Lou, and C.-Y. Chen, 2009. A Low-Complexity LUT-Based Squaring Algorithm, Computers & Mathematics with Applications, 57, s. 1494-1501.

6. S. Gao, N. Chabini, D. Al-Khalili, and P. Langlois, 2007. Optimised Realisations of Large Integer Multipliers and Squarers Using Embedded Blocks, IET Computers & Digital Techniques, 1, s. 9-16.

7. J. L. Athow and A. J. Al-Khalili, 2008. Implementation of Large-Integer Hardware Multiplier in Xilinx FPGA, in Electronics, Circuits and Systems, 2008. ICECS 2008. 15th IEEE International Conference on, s. 1300-1303.

8. S. Xu, S. A. Fahmy, and I. V. McLoughlin, 2013. Efficient Large Integer Squarers on FPGA, in Field-Programmable Custom Computing Machines (FCCM), 2013 IEEE 21st Annual International Symposium on, s. 198-201.

9. A. Senturk and M. Gok, 2012. Pipelined Large Multiplier Designs on FPGAs, in Digital System Design (DSD), 2012 15th Euromicro Conference on, s. 809-814.