İleri Teknoloji Bilimleri Dergisi
Cilt 2, Sayı 2, 1-13, 2013

Journal of Advanced Technology Sciences
Vol 2, No 2, 1-13, 2013

FPGA TABANLI ORTALAMA GÖRÜNTÜ FİLTRESİ TASARIMI

Süleyman ÇAKICI[1]*, İbrahim ŞAHİN[2]

[1] Düzce Üniversitesi, Teknoloji Fakültesi, Bilgisayar Mühendisliği Bölümü, 81620, Düzce, TÜRKİYE,
[2] Düzce Üniversitesi, Teknik Eğitim Fakültesi, Elektronik ve Bilgisayar Eğitimi Bölümü, 81620, Düzce, TÜRKİYE

Özet-Literatürde, görüntü kalitesini artırmak için kullanılan birçok iyileştirme tekniği bulunmaktadır. Bunlardan biri olan ve günlük uygulamalarda oldukça yaygın olarak kullanılan görüntü filtreleme işlemi, yazılım tabanlı olarak gerçekleştirildiğinde, oldukça fazla CPU zamanı anlamına gelmektedir. Bu çalışmada, görüntü filtreleme işlemini hızlandırmak amacıyla FPGA çipleri ile kullanılabilecek bir donanım modülü tasarlanmıştır. Tasarlanan modül, gerçek veriler üzerinde test edilerek modülün ürettiği sonuçların doğrulaması yapılmıştır. Modülün sonuçları ile farklı özelliklere sahip bilgisayarlar üzerinde gerçekleştirilen yazılım tabanlı uygulama sonuçları karşılaştırılarak, modülün veri işleme başarımı incelenmiştir. Karşılaştırma sonuçları, tasarlanan modülün görüntü filtreleme işlemlerini genel amaçlı bilgisayarlara nispeten yaklaşık 2 ile 5 kata kadar daha hızlı gerçekleştirilebildiğini göstermiştir.

Anahtar Kelimeler- FPGA, Görüntü Filtreleme, Ortalama Filtre, Donanım Modülü.

FPGA BASED MEAN IMAGE FILTER DESIGN

Abstract-There exists a wide variety of techniques for improving image quality in the literature. The image filtration is a widely used image enhancement technique in modern day applications. When performed through software, it takes considerable amount of CPU time especially for larger images. In this study, a hardware module that can be used on FPGA chips was designed to speed up image filtration process. The module was tested using test data and functional verification is done by comparing module's results with a software implementation's results. Module's performance was evaluated by comparing its timing results with the timing of software implementation running on two different general purpose computers. The results showed that the depending on the size of the image, the Module can perform filtration between about 2 to 5 times faster than the software implementation running on general purpose computers.

Key Words- FPGA, Image Filter, Mean Filter, Hardware Module.

1. GİRİŞ (INTRODUCTION)

Son yıllarda, gerçek zamanlı ve yüksek başarım gerektiren işaret işleme uygulamalarının gerçeklemesinde FPGA (Alan Programlanabilir Kapı Dizileri) kullanımının arttığı

* suleymancakici@duzce.edu.tr

görülmektedir. Başlangıçta sayısal tasarımların test edilmesi amacıyla geliştirilen FPGA çipleri, FPGA teknolojisindeki güncel gelişmelere paralel olarak, yüksek dereceli paralel çalışabilme kabiliyetine sahip olmuştur [1, 2].

Görüntü filtreleme, işaret işlemenin bir türüdür ve günümüzde oldukça yaygın olarak kullanılmaktadır. Filtreleme işlemleri yazılım tabanlı olarak gerçekleştirildiğinde, özellikle büyük boyutlara sahip görüntüler için, oldukça fazla CPU zamanı anlamına gelmektedir. Söz konusu problemin çözümüne ilişkin değişik yaklaşımlar geliştirilmiştir. Bu yaklaşımlar; gelişmiş grafik kartlarının, grafik işlemler için tasarlanmış özel amaçlı bilgisayarların, daha hızlı ya da paralel çalışan işlemcilerin kullanılması olarak özetlenebilir. Bununla birlikte, sunulan çözümler bazı sakıncaları da beraberinde getirmektedir. Örneğin, kişisel bilgisayarlar grafik işlemlerinin gerçekleştirilmesinde yeterli performans gösteremezken, süper bilgisayarlar yüksek maliyetli çözümlerdir. Uygulamalara özel olarak tasarlanan kartlarda (ASIC), karşılaşılan herhangi bir tasarım veya üretim hatasının telafi edilmesi imkânsız olmakla birlikte, yeniden tasarım süreci oldukça fazla zaman kaybına sebep olmaktadır [3].

Görüntü filtreleme işlemlerini hızlandırmak amacıyla, literatürde sunulan birçok çalışma bulunmaktadır. P. Nirmal Kumar ve dig., işlenecek resmin türüne göre kendini güncelleyebilen bir donanım tasarlanmıştır [4]. Geleneksel yaklaşımlarda, tasarım sürecinde geri dönüşümü olmayan sabit tasarım tekniği kullanıldığı için, tasarlanan sistem filtreleme işlemine esneklik kazandırmıştır. Tasarım, üzerine Gaussian gürültüsü eklenmiş, değişik varyanslardaki 128×128'lik görüntülere uygulanmıştır. Kaijun Tang ve dig. gürültü sebebiyle zarar görmüş renkli görüntülerin düzeltilmesi üzerine uyarlanabilir karma bir filtre tasarımı yapılmıştır [5]. Tasarlanan filtrenin başarımı, uyarlanabilir olmayan diğer filtreler ile karşılaştırılmış; renkli görüntü örneklerinde, uyarlanabilir karma filtrelerin daha iyi bir başarım ortaya koyduğu görülmüştür. Diğer bir çalışmada, 3×3 pencere boyutuna sahip Kapasitif Eşik Mantığı (CTL) teknolojisinin kullanıldığı yeni bir 2D görüntü filtre mimarisi tasarlanmıştır [6]. Tamamen ardışık ve paralel mimari kullanılarak oluşturulan bu filtre, ASIC olarak modellenmiş ve benzetimi yapılmıştır. Benzetim sonuçları göstermiştir ki; eğer devre ASIC olarak gerçekleştirilirse 100 MHz de çalışabilmekte ve 1024×1024 boyutunda görüntülerde saniyede 50 görüntü işleyebilecek kapasitededir. Burada ASIC tasarımın hız avantajı kullanılmıştır. Y. Yano ve dig. ayarlanabilir bir filtre sistemi geliştirmiştir [7]. Çalışmada, ikili (binary) görüntüler üzerinde çalışılmış, istenilen filtre seti 3×3'lik ikili matrislerin kromozom olarak kabul edildiği genetik algoritma ile oluşturulmuştur. Önerilen sistemin etkililiğini göstermek için benzetimler yapılmış ve FPGA ile yapılan uygulamanın yazılımdan 48,6 kat daha hızlı olduğu görülmüştür.

Bu makalede sunulan çalışmada; görüntü filtreleme işlemini hızlandırmak amacıyla, yukarıda ifade edilen çözümlere alternatif olarak, FPGA çipleri üzerinde çalışabilecek bir görüntü filtreleme donanım modülü tasarlanmıştır. Böylece, görüntü filtrelemede daha ucuz, daha hızlı ve FPGA çiplerinin tekrar programlanabilme özellikleri sayesinde daha esnek bir yapının ortaya koyulması hedeflenmiştir. Tasarlanan modül, gerçek veriler üzerinde işlemler yapılarak test edilmiş ve modülün ürettiği sonuçların doğrulaması yapılmıştır. Aynı veriler, değişik özellikteki genel amaçlı bilgisayarlar üzerinde de işlenmiştir. Elde edilen sonuçlar kullanılarak, modülün veri işleme hızı bilgisayarlarla karşılaştırılmıştır.

Makalenin ikinci bölümünde, öncelikle görüntü filtrelemede kullanılan Ortalama Filtre'nin (Mean Filter) matematiksel modeli sunulmuştur. Ardından, tasarlanan modül detaylarıyla anlatılmıştır. Üçüncü bölümde ise, gerçekleştirilen test çalışmaları ve elde edilen sonuçlar verilmiştir. Elde edilen sonuçlar son bölümde değerlendirilmiştir.

2. GÖRÜNTÜ FİLTRELEME İÇİN FPGA MODÜL TASARIMI (DESINGING A FPGA MODULE FOR IMAGE FILTERING)

Güncel uygulamalarda, görüntü filtreleme için önerilen birçok yöntem bulunmaktadır. Bu çalışmada Ortalama Filtre (Mean/Avarage Filter) yöntemi kullanılmıştır. Ortalama filtre, görüntüleri iyileştirmek ya da önceki ve sonraki pikseller arasındaki yoğunluk değişim miktarını azaltmak için kullanılan temel ve kolay bir yöntemdir. Genellikle görüntünün içindeki gürültüyü (bozukluk) azaltmak için kullanılmaktadır [8].

Bir görüntüye ortalama filtre yöntemi uygulanırken, görüntüdeki her bir piksel değerinin, o piksele komşu diğer piksel değerleriyle birlikte ortalaması alınır. Seçilen pikselin değeri, elde edilen yeni değer ile değiştirilir. Örneğin; *P(x,y)* fonksiyonu, Şekil 1'de görülen *x,y* konumundaki piksel değerini versin. Ortalama Filtre uygulanmış yeni görüntünün *x,y* konumundaki piksel değeri ise *N(x,y)* olsun.

x-2, y-2	x-1, y-2	x, y-2	x+1, y-2	x+2, y-2
x-2, y-1	x-1, y-1	x, y-1	x+1, y-1	x+2, y-1
x-2, y	x-1, y	x, y	x+1, y	x+2, y
x-2, y+1	x-1, y+1	x, y+1	x+1, y+1	x+2, y+1
x-2, y+2	x-1, y+2	x, y+2	x+1, y+2	x+2, y+2

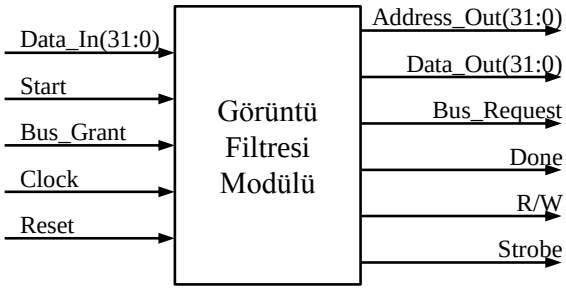
Şekil 1. Ortalama filtrede genel olarak kullanılan 5×5'lik çekirdek (The 5×5 core that is commonly used in mean filter)

Bu durumda, filtrelenecek görüntüdeki her bir piksel için N(x,y) fonksiyonu Eş.1'deki gibi hesaplanır:

$$N(x,y)=\frac{1}{25}\begin{pmatrix} P(x-2,y-2)+P(x-1,y-2)+P(x,y-2)+P(x+1,y-2)+P(x+2,y-2)+ \\ P(x-2,y-1)+P(x-1,y-1)+P(x,y-1)+P(x+1,y-1)+P(x+2,y-1)+ \\ P(x-2,y)+P(x-1,y)+P(x,y)+P(x+1,y)+P(x+2,y)+ \\ P(x-2,y+1)+P(x-1,y+1)+P(x,y+1)+P(x+1,y+1)+P(x+2,y+1)+ \\ P(x-2,y+2)+P(x-1,y+2)+P(x,y+2)+P(x+1,y+2)+P(x+2,y+2) \end{pmatrix} \quad (1)$$

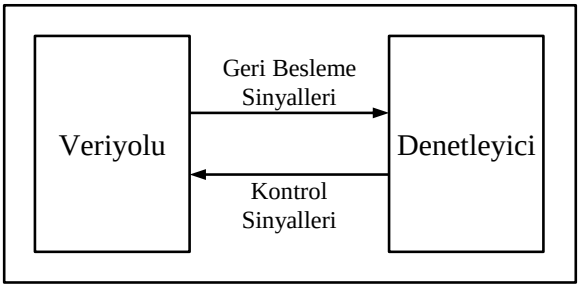
Filtreleme işlemi sonucunda, filtre uygulanan bölgedeki düzgün olmayan, bozulmaya uğramış piksel değerleri elenmiş olur. Ortalama filtre yönteminde genellikle küçük boyuttaki çekirdekler (3×3'lük kare çekirdek gibi) kullanılmasına rağmen, görüntünün daha fazla düzgünleştirmesini istendiği durumlarda Şekil 1'de gösterilen 5×5'lik kare çekirdek kullanılır. Ayrıca, resmin küçük boyuttaki çekirdek ile bir defadan fazla filtrelenmesi ile de büyük boyuttaki bir çekirdeğin etkisine yakın bir etki oluşturulabilmektedir.

Bu çalışmada, FPGA tabanlı ortamlarda kullanılabilecek bir görüntü filtresi modülü tasarlanmıştır. Tasarlanan modülün en üst seviye blok diyagramı Şekil 2'de gösterilmiştir. Modül, 32 bitlik `Data_In` girişi ile birer bitlik `Start`, `Bus_Grant`, `Clock`, `Reset` girişlerine sahiptir. Ayrıca, 32 bitlik `Address_Out` ile `Data_Out` çıkışları ile birer bitlik `Bus_Request`, `Done`, `R/W` ve `Strobe` çıkışları bulunmaktadır. `Reset`, `Başla` ve `Bitti` sinyalleri modülün zamanlaması ve bağlı oldukları bilgisayar arasındaki zaman eşlemesini sağlamak için kullanılır. `Bus_Grant` ve `Bus_Request` sinyalleri ise, hafıza ile olan zaman eşlemesini sağlamak ve veri okuma yazma işlemleri için kullanılır. Tasarlanan modül, bir donanım tanımlama dili olan VHDL'de kodalanmış ve Xilinx ISE 10.1 yazılımı kullanılarak değişik FPGA çiplerine göre sentezlenmiştir.



Şekil 2. Önerilen filtre modülünün blok diyagramı (The block diagram of the proposed filter module)

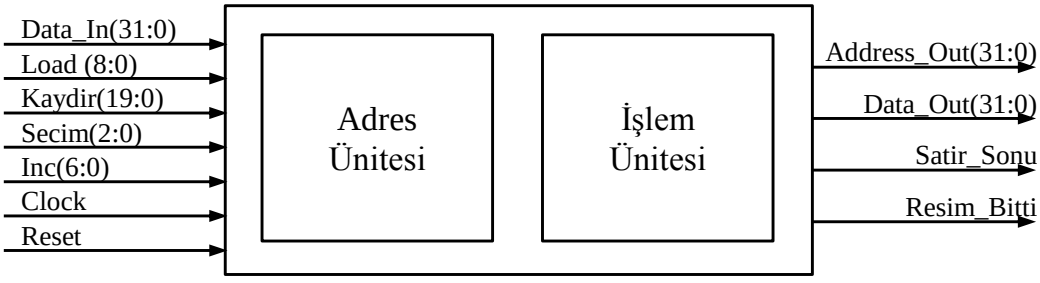
Filtre modülünün ikinci seviye blok diyagramı Şekil 3'te gösterilmiştir. Alt-modül, *Veriyolu* ve *Denetleyici* olmak üzere iki blok halinde tasarlanmıştır. *Veriyolu* bloğunda, adres hesaplamaları ve filtreleme için gerekli matematiksel işlemler yapılmaktadır. *Denetleyici* bloğunda ise, yukarıda bahsedilen işlemlerin zamanını, sırasını ve yapılacağı süreyi bildiren ve veriyolunu denetleyen sinyaller üretilmektedir.



Şekil 3. Filtre modülünün ikinci seviye blok diyagramı (The second-level block diagram of the filter module)

2.1. Veriyolu Bloğu (The DataPath Block)

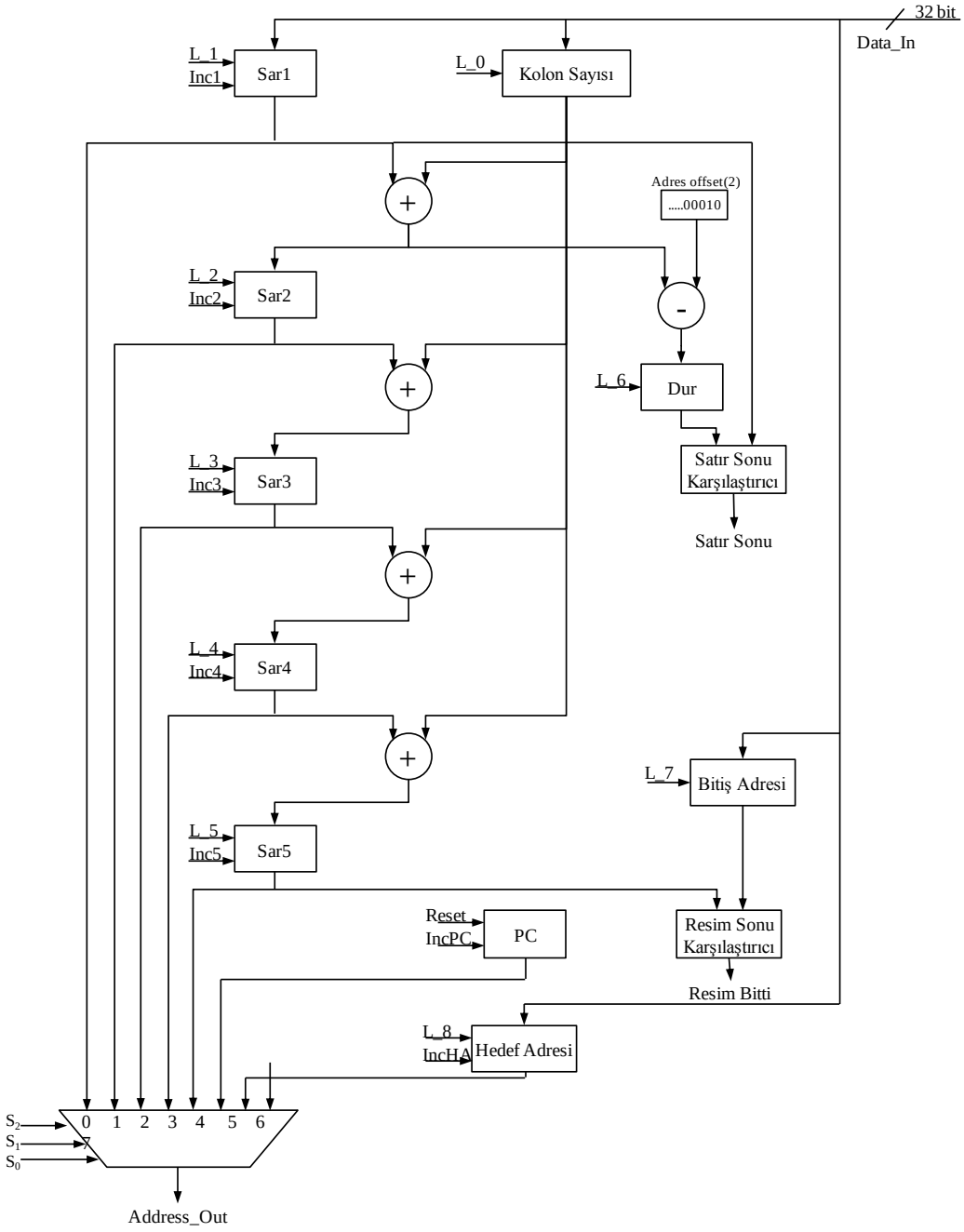
Veriyolu bloğunda, hafızaya erişim için gerekli adres hesaplamaları ve hafızadan alınan görüntü bilgisinin filtrelenmesi için gerekli matematiksel işlemler yapılmaktadır. Adres ve İşlem olarak adlandırılan iki alt üniteden oluşmaktadır. `Data_In`, `Load`, `Kaydir`, `Secim`, `Inc`, `Clock`, `Reset` giriş sinyalleri ile; `Address_Out`, `Data_Out`, `Satir_Sonu`, `Resim_Bitti` çıkış sinyalleri bulunmaktadır. Şekil 4'te *Veriyolu*'nun blok diyagram görülmektedir.



Şekil 4. Veriyolu blok diyagramı (The block diagram of the DataPath)

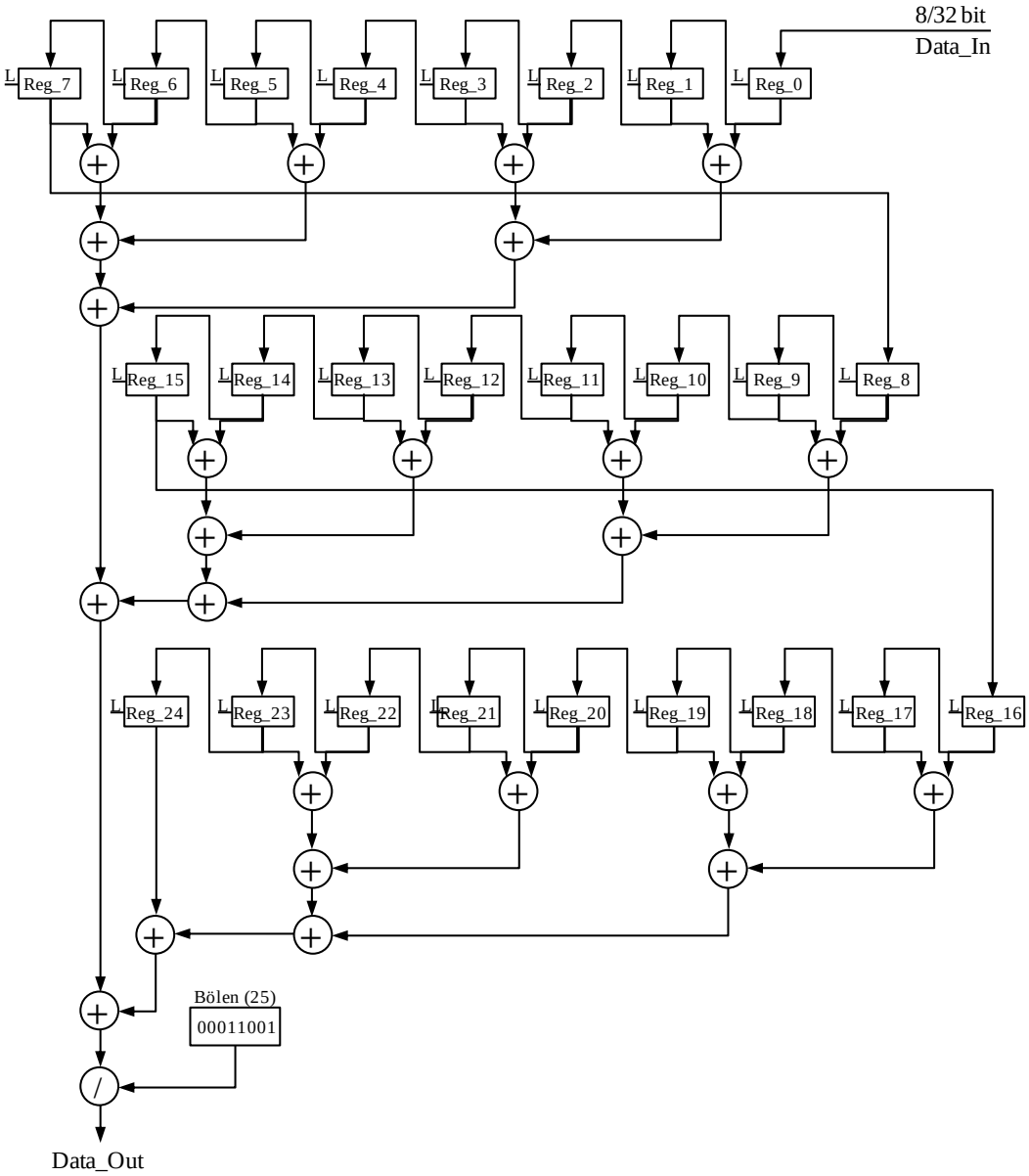
Adres ünitesi *kaydedici*, *multiplexer*, *toplayıcı* ve *karşılaştırıcı* ünitelerinden oluşmaktadır. *Toplayıcı* üniteleri için özel bir tasarım yapılmamış, VHDL sentez aracının bu üniteleri sentezlemesi sağlanmıştır. Şekil 5'te Adres ünitesinin detaylı blok diyagramı görülmektedir. *Sar1, .. Sar5* kaydedicileri hafızadaki resim bilgisinin filtreleme yapılan geçerli satır adreslerini

tutarlar. `Load` sinyali, kaydedicilerin içine bilgi okumak için, `Inc` sinyali de kaydedicilerin içeriğini 1 artırmak içindir. *Kolon Sayısı*, *Hedef Adresi*, *Bitiş Adresi* kaydedicilerine `Load` sinyali ile birlikte başlangıç parametreleri kaydedilir. PC, görüntü parametrelerinin okunduğu sırada hafıza adreslerini takip etmek için kullanılmıştır. Parametreler okunurken `Inc` sinyali ile içeriği 1 artırılır; parametreler okunduktan sonra `Reset` sinyali ile sıfırlanır. *Multiplexer* ünitesi, seçme bitlerine gelen bilgiye göre ilgili adres kayıcısını `Address_Out` çıkışlarını gönderilmesini sağlar. *Karşılaştırıcı* üniteler, girişlerine uygulanan bilgiler eşit olduğunda "1" çıkışı verirler. Bu üniteler bir satırın işlenip bitiğinin belirlenmesinde ya da resmin tamamının işlenip bittiğinin belirlenmesinde kullanılırlar.



Şekil 5. Adres Ünitesi'nin detaylı blok diyagramı (The detailed block diagram of the Address Unit)

İşlem ünitesinde (Şekil 6) 8 bitlik yirmi beş adet kaydedici (Reg0-Reg24), değişik büyüklüklerde 24 adet toplayıcı ve bir bölücü üniteden oluşmaktadır. Toplayıcı üniteler için özel bir tasarım yapılmamış, VHDL sentez aracının bu üniteleri sentezlemesi sağlanmıştır. Bununla birlikte, bölme ünitesi 1 bitlik AddSub hücreleri kullanılarak tasarlanmıştır.

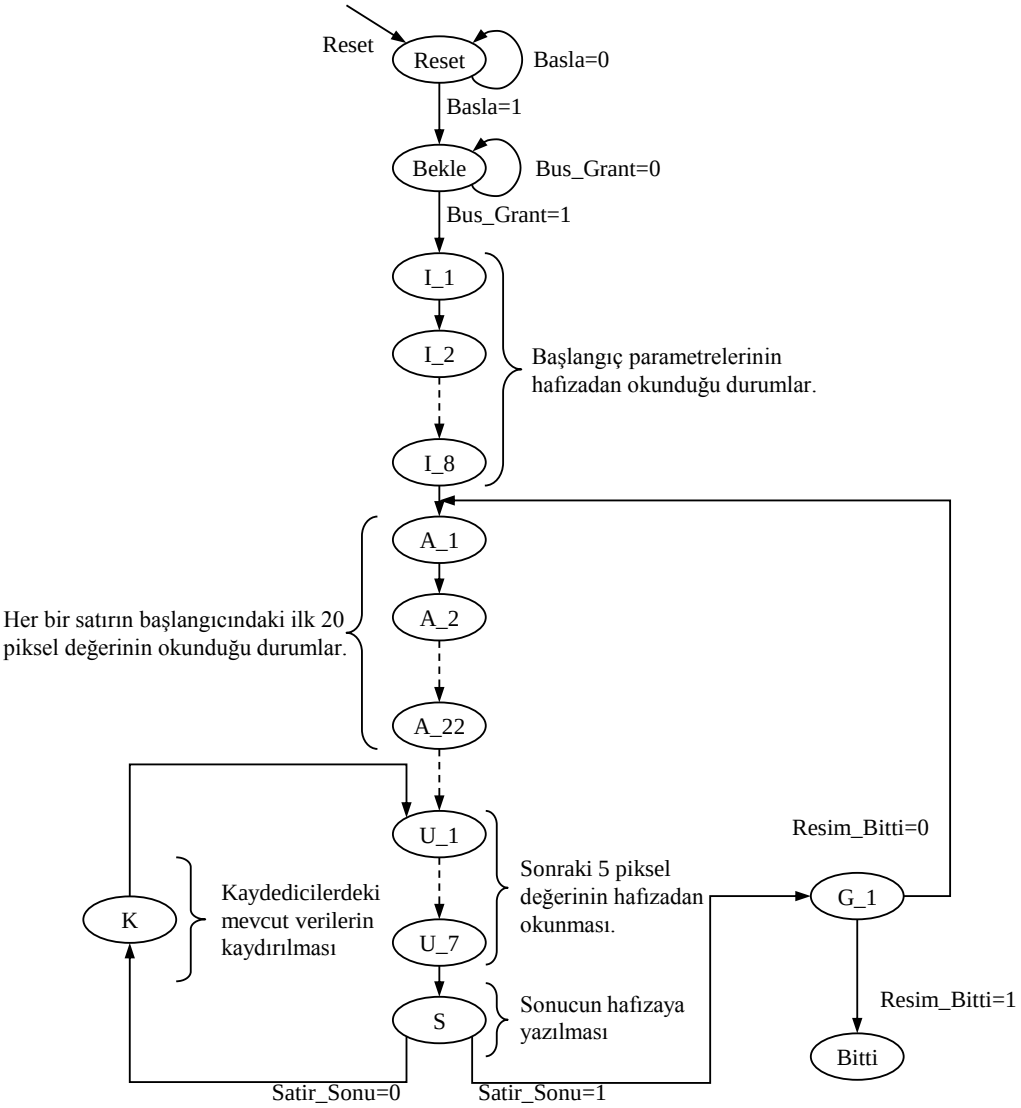


Şekil 6. İşlem ünitesinin detaylı blok diyagramı (The detailed block diagram of the Processing Unit)

Kayıtcılar birbirine seri olarak bir zincir oluşturacak şekilde bağlanıştır. Bu şekilde bağlantı kayıtçıların daha kolay kontrol edilmesini sağlamakta ayrıca hafızadan okunan bilgilerin kayıcılara aktarılmasında da herhangi bir dezavantaj oluşturmamaktadır. L (`Load`) sinyali, tüm kaydediciler için ortak bağlantıya sahiptir. Her L sinyali alındığında, kaydedicilerdeki mevcut veriler bir sonraki kaydedicilere aktarılır ve `Data_In` üzerindeki 8 bitlik veri, Reg_0 kaydedicisine yazılır. Adres bloğundaki adresler doğrultusunda, ilgili adreslerdeki resim bilgisi kaydedicilere kaydedilmesinin ardından, gerekli toplama ve bölme işlemleri yapılır. `Data_Out` çıkışından alınan sonuç hafızaya kaydedilmektedir.

2.2. Denetleyici Bloğu (The Controller Block)

Tasarlanan filtre modülünün denetleyicisi toplam 44 durumdan oluşan bir Sonlu Durum Makinesi (Finite State Machine-FSM) olarak tasarlanmıştır. Denetleyicinin durum diyagramı Şekil 7'da gösterilmiştir. Modül ilk çalışmaya başladığında *Reset* durumunda bekler. `Basla` sinyalini aldığında, Reset durumundan çıkarak hafızaya erişmek için sistem yoluna istek sinyali gönderir ve yolun kendine tahsis edilmesini bekler.



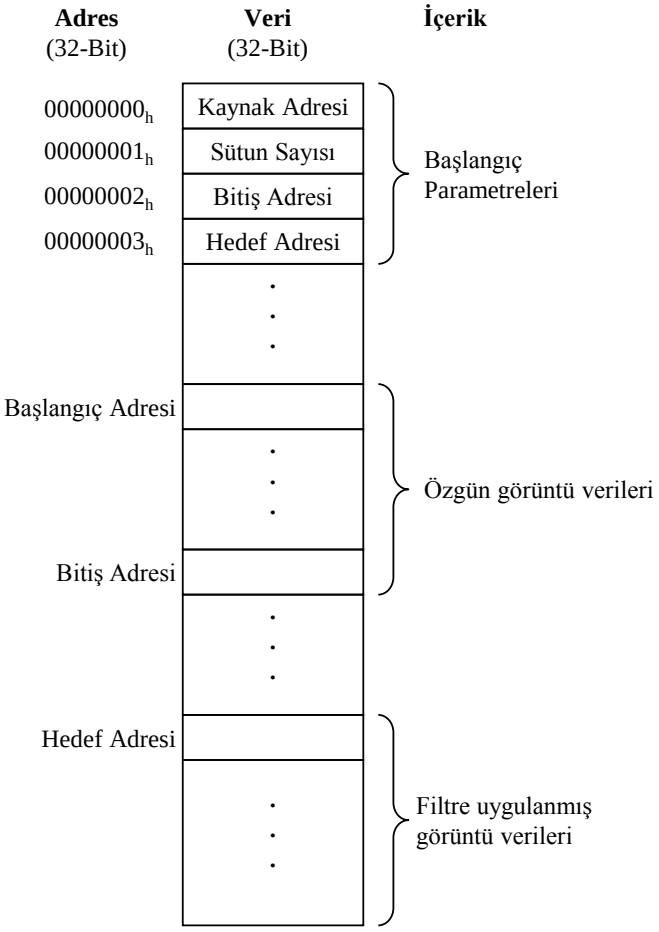
Şekil 7. Görüntü filtresine ait denetleyicinin durum diyagramı (The controller state diagram of the proposed image filter module)

Yol tahsis edildikten sonra, hafızadaki `0…0h` adresine gidilerek *başlangıç adresi*, *kolon sayısı*, *bitiş adresi* ve *hedef adresi* alınır. Bu bilgiler, Adres bloğu içindeki uygun kaydedicilere yüklenir. Adres bilgilerinin okunması sırasında biyandanda, adres bloğunda bulunan satırları adreslemede kullanılan Sar2 ... Sar5 değerleri hesaplanarak ilkili kayıtçılarak kaydedilir. Bu işlemler, denetleyicide sekiz durum boyunca yapılır. Altı adet adres okunmasına rağmen söz konusu işlemlerin sekiz durum boyunca sürmesinin nedeni, okuma işlemlerinde verinin hafızadan, adres gönderildikten 2 saat darbesi sonra okunabilmesidir. Okunan bu değerler bütün noktaların dönüşümü tamamlanıncaya kadar bu yazmaçlar içinde tutulur. Modül, başlangıç

değerlerini aldıktan sonra, başlangıç adresinden itibaren hafızadan resim bilgisini okumaya başlar. İlk olarak geçerli 5×5 lik çerçeve içine denk gelen, resmin ilk 20 piksel değeri okunur. Böylece, çerçeve içinde bir sütun 5 pikselden oluştuğu için, ilk 4 sütun okunmuş olur. Bu işlem 22 durum boyunca devam eder. Son sütun değerinin alınmasının ardından, okunan toplam 25 pikselin ortalaması otomatik olarak hesaplanır. Elde edilen sonuç, hedef adresinin gösterdiği hafıza alanına yazılır (S durumu). Bu işlem 7 durum boyunca sürdürülür. Resmin ilk satırındaki değerler taranıp alt satıra geçilene kadar (`Satir_Sonu=1` oluncaya kadar), her defasında resmin 5×5 lik çerçeve içinde denk gelen ikinci sütunundaki değerler birinci sütununa, üçüncü sütunundaki değerler ikinci sütuna, dördüncü sütundaki değerler üçüncü sütuna ve en son beşinci sütundaki değerler ise dördüncü sütuna kaydırılır (K Durumu). Bu kaydırma işlemi sayesinde herhangi bir pikselde daha öncesindeki bir piksel için hafızadan okunmuş ve ortak olan 20 değer aynen kullanılır ve hafıza erşiminde tasarruf sağlanır. Eğer, satır sonuna gelinmişse, G-1 durumunda bir alt satıra geçilerek satır başı işlemi yapılır ve çerçeve içindeki ilk 4 sütun değeri ve son sütun değerleri okunur. Resmin sonuna gelindiğinde (`Resim_Bitti=1` olunca) ise, işlemler sonlandırılır ve tüm kaydediciler sıfırlanır. Sistem yeniden başa döndürülür ve Başla sinyalinin gelmesi beklenir.

2.3. Hafıza Haritası (The Memory Map)

Şekil 8'da tasarlanan filtre modülünün kullandığı hafıza haritası görülmektedir. Hafızada adreslenebilir her bir veri alanının 32-bit olduğu varsayılmıştır.



Şekil 8. Filtre modülüne ait hafıza haritası (The memory map of the proposed filter module)

Hafıza haritası üç bölümden oluşmaktadır. Birinci bölüm, başlangıç değerlerinin tutulduğu hafıza alanıdır. Hafıza haritasının `0…0h` ve `0…3h` adresleri arasındaki, her biri 32-bitlik dört adet hafıza gözünü içerir. `0…0h` adresinde filtrelenecek görüntü bilgisinin başlangıç adresi (*Sar1*), `0…1h` adresinde filtrelenecek görüntünün sütun sayısı, `0…2h` adresinde görüntü bilgisinin bitiş adresi saklanır. Yapılan hesaplamalar sonucunda elde edilen yeni değerlerin hafızada hangi adresten itibaren yazılmaya başlanacağını gösteren adres bilgisi *Hedef Adresi* adı altında `0…3h` adresinde tutulur. İkinci bölüm, hafızada *Sar1* adresi ile başlayan ve görüntünün orijinal değerlerinin kayıtlı olduğu hafıza alanıdır. Bu alanlara yazılacak değerler modülü kullanmak üzere geliştirilecek ara yüz programı tarafından hesaplanarak yazılmalıdır. Üçüncü bölüm, Bitiş Adresi ile başlayan, hesaplanan yeni değerlerin yazıldığı hafıza alanıdır. Resmin orijinal görüntü boyutu ve hedef alanın boyutu işlenecek görüntünün boyutuna göre değişmektedir.

2.4. Görüntü Filtreleme Yöntemi (The Method of Image Filtering)

Bu çalışmada, filtrelenecek görüntünün en dışında kalan çerçeve piksellerin filtrelemesi yapılmamıştır. Şekil 9'daki örnekte görüldüğü gibi, görüntünün çerçeve pikselleri hariç, kırmızı çizgi ile belirtilen sınırlar içerisindeki tüm pikseller için ortalama filtre uygulaması yapılmıştır. İki piksellik çerçeve alan küçük resimlerde önemli olmasına karşın büyük resimlerde örneğin 1024x1024 resmin %1'in den daha küçük bir alanına denk gelmektedir ve önemsizdir.

0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	0,10	0,11
1,0	1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8	1,9	1,10	1,11
2,0	2,1	2,2	2,3	2,4	2,5	2,6	2,7	2,8	2,9	2,10	2,11
3,0	3,1	3,2	3,3	3,4	3,5	3,6	3,7	3,8	3,9	3,10	3,11
4,0	4,1	4,2	4,3	4,4	4,5	4,6	4,7	4,8	4,9	4,10	4,11
5,0	5,1	5,2	5,3	5,4	5,5	5,6	5,7	5,8	5,9	5,10	5,11
6,0	6,1	6,2	6,3	6,4	6,5	6,6	6,7	6,8	6,9	6,10	6,11
7,0	7,1	7,2	7,3	7,4	7,5	7,6	7,7	7,8	7,9	7,10	7,11
8,0	8,1	8,2	8,3	8,4	8,5	8,6	8,7	8,8	8,9	8,10	8,11

Şekil 9. Örnek görüntü bilgisi piksel koordinatları (The pixel coordinates of the example image information)

Hafızadan, öncelikle başlangıç adresi (Sar1), kolon sayısı, bitiş adresi ve hedef adresi okunur. Okunan Sar1'den itibaren değerler sırayla hafızadan alınır. Sar1 ((0,0) indisli piksel) adresi okunduktan hemen sonra, Sar1'e kolon sayısı eklenerek Sar2 ((1,0) indisli piksel) adresi hesaplanır. Benzer şekilde Sar3 ((2,0) indisli piksel), Sar4 ((3,0) indisli piksel) ve Sar5 ((4,0) indisli piksel) adresi hesaplanır.

Bir pikselin filtre işlemi yapılırken, Sar sayaçlarında tutulan adresler sırasıyla Adres Bus'a yazılarak, ilgili hafıza adreslerindeki piksel değerleri okunur ve işlem ünitesi içindeki kayıtçılara aktarılır. Ardından, Sar1, Sar2, Sar3, Sar4 ve Sar5 sayaçlarının değerleri "1" artırılarak 5×5 çerçevede bir sonraki sütüna geçilir. Bu işlem, toplam 5 defa tekrar edilerek 5×5 çerçeve içindeki tüm piksellerin işlem ünitesi içindeki kayıtçılara aktarılması sağlanır. 5×5'lik çerçece

içindeki 25 değerin tamamının hafızadan alınmasının ardından, değerlerin ortalaması hesaplanır ve elde edilen sonuç hafızada hedef adrese yazılır. Böylece 2,2 indisli piksel için filtreleme yapılmış olur. Bir sonraki piksel için yine Sar sayaçları "1" artırılır, 5×5 çerçevede bu defa sadece son sütün okunur. Çünkü bir önceki piksel için okunan son 4 sütun, şu anda geçerli olan piksel için ilk dört sütunu oluşturmaktadır ve bu değerler hali hazırda işlem ünitesi içinde kayıtçılarda bulunmaktadır. Bu durum hafıza erişimi açısından önemli bir avantaj oluşturmaktadır. Piksellerin bu şekilde işlenmesi Sar1 sayacının değeri Dur sayacına eşit olana kadar devam eder. Sar1 ve Dur sayaçlarının eşit olması, ilgili satırın işlenmesinin bittiği anlamına gelir ve denetleyici satır başına gitmek için gerekli işlemleri yapar.

3. TASARLANAN FİLTRE MODÜLÜNÜN KARŞILAŞTIRMALI TEST SONUÇLARI (THE COMPARATIVE TEST RESULTS OF DESIGNED FILTER MODULE)

Bu çalışmada tasarlanan filtre modülü, Xilinx firması tarafından üretilen *Virtex5* ve *Virtex4* çipleri için sentezlenerek, modülün FPGA çip istatistikleri ve maksimum saat frekansları incelenmiştir. Modülün belirlenen bir görüntü verisini işleme süresi, ISE benzetim programı kullanılarak elde edilmiş ve farklı PC'lerin aynı veriyi işleme süreleri ile karşılaştırılmıştır. Tablo 1'de modüle ait sentezleme sonuçları görülmektedir. Sentezleme sonuçları göstermiştir ki, kullanılan slice register, LUTs (Look-Up Tables) sayılarına göre, teorik olarak, modülün *Virtex4* çipine on, *Virtex5* çipine yirmi adet kopyasının aynı anda sığması mümkündür. Fakat kullanılan IOB (giriş/çıkış pinleri) dikkate alındığında, *Virtex4* ve *Virtex5* çipine yalnız iki adet modülün rahat bir şekilde yerleştirilebileceği görülmektedir.

Tablo 1. Sentezlenen filtre modülünün FPGA çip istatistikleri (The FPGA chip statistics of synthesized filter module)

FPGA Çip Türü	Slice Reg. Sayısı / %	LUTs Sayısı / %	Slice FFs Sayısı / %	Bounded IOBs Sayısı / %
Virtex5	537 / 02	1047/ 05	530 / 03	110 / 50
Virtex4	525 / 03	1201 / 09	565 / 04	110 / 45

Modülün çalışma performansını karşılaştırmak için özellikleri Tablo 2'de verilen üç adet bilgisayar seçilmiştir. Seçilen bu bilgisayarlardan PC-1 tek çekirdekli AMD işlemciye, PC-2 tek çekirdekli Intel işlemciye, PC-3 çift çekirdekli AMD işlemciye sahiptir.

Tablo 2. Deneylerde kullanılan bilgisayarların özellikleri (The features of the computers used in the tests)

PC Adı	CPU Hızı (GHz)	CPU Ön Bellek (KB)	RAM Hafıza Boyutu (MB)	Hafıza Tipi	CPU Türü	FSB BUS Hızı (MHz)	BUS Boyutu (Bit)
PC-1	1.6	512	1024	DDR 1	AMD Turion	400	32
PC-2	1.73	2048	512	DDR 1	Intel Pentium	366	32
PC-3	1.66	2048	512	DDR 2	AMD	533	64

Deneylerde kullanılan farklı boyutlardaki görüntülere ait veriler, bir C program parçacığı kullanılarak oluşturulmuştur. Bu verileri tutan diziler üzerinde ortalama filtre yöntemi uygulanmış ve sonuçlar tekrar görüntü bilgisini tutan bu dizilere aktarılmıştır.
Tablo **3**'te PC'lerin söz konusu verileri işleme süreleri mikro saniye (µs) türünden verilmiştir.
Tablo **4**'te ise *Virtex5* ve *Virtex4* çiplerinin deney verilerini işleme süreleri verilmiştir.

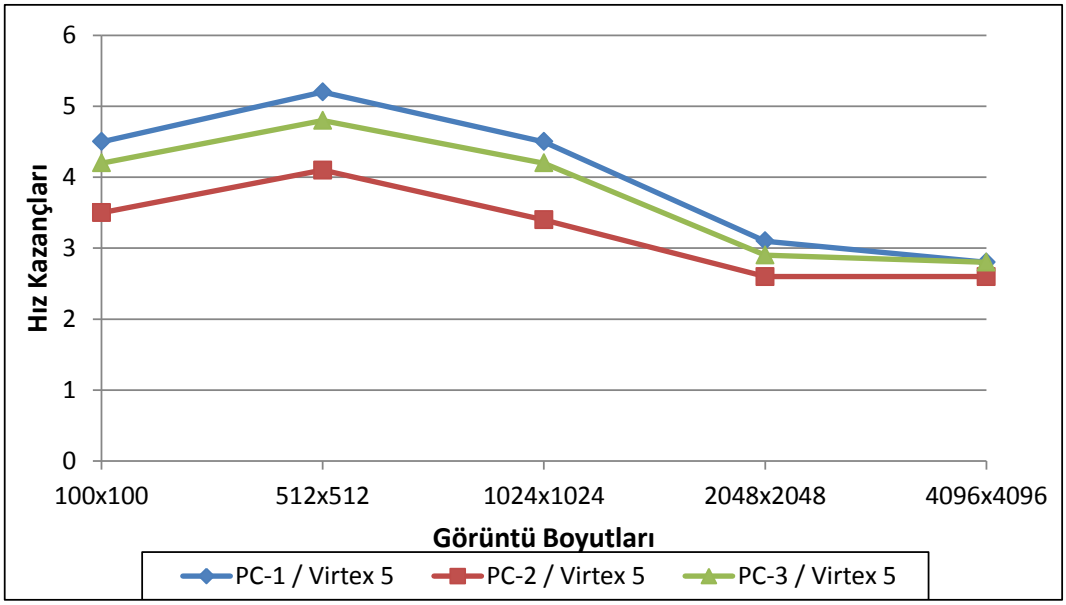
Tablo 3. PC'lerin farklı boyutlardaki görüntü verilerini işleme süreleri (PCs' processing times of different sized image data)

Görüntü Boyutu	PC-1 (*µs*)	PC-2 (*µs*)	PC-3 (*µs*)
100×100	1568.775	1217.934	1453.850
512×512	44337.921	34597.118	41266.737
1024×1024	161496.688	121317.550	148561.093
2048×2048	435080.743	375909.625	413145.168
4096×4096	1623000.137	1519816.019	1577064.562

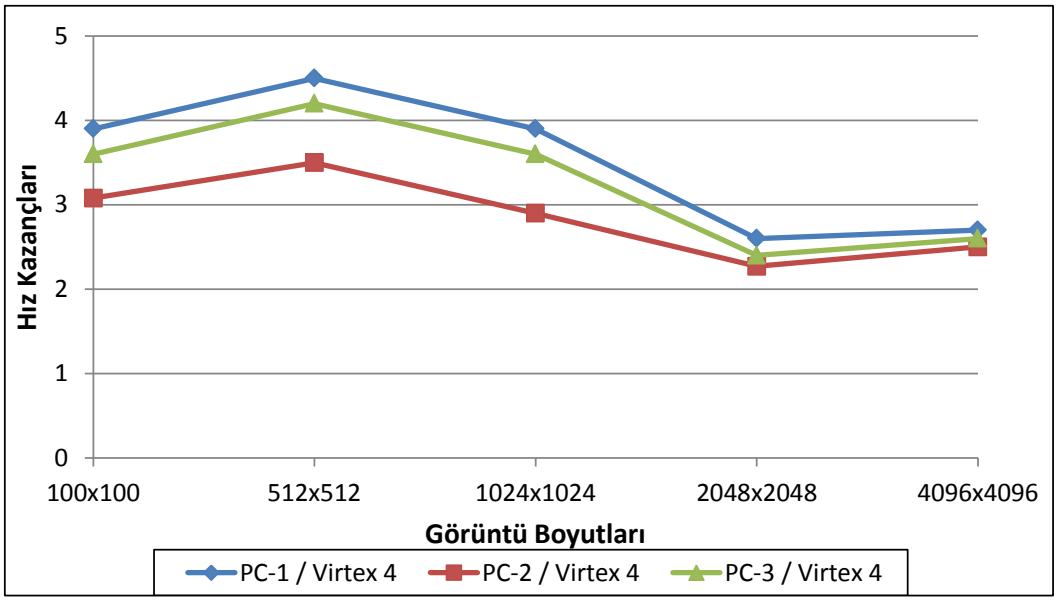
Tablo 4. Seçilen FPGA çiplerinin farklı boyutlardaki görüntü verilerini işleme süreleri (FPGA chips' processing times of different sized image data)

Nokta Sayısı	*Virtex5* (*µs*)	*Virtex4* (*µs*)
100×100	343.250	394.43
512×512	8426.116	9791.976
1024×1024	35258.775	40733.555
2048×2048	140030.330	165505.18
4096×4096	563191.698	588666.478

Şekil 10 ve Şekil 11'te, sırasıyla modülün *Virtex5* ve *Virtex4* çiplerine üzerindeki çalışma sürelerinin, PC sonuçları ile karşılaştırılması görülmektedir. Her bir grafikte, farklı boyutlardaki görüntü verileri için, bir FPGA çipine göre sentezlenmiş modülün PC'lere göre hız kazancı görülmektedir. Virtex 5'te modül, resim boyutuna göre değişmekle birlikte, PC-1'e göre yaklaşık 5, PC-2'ye göre yaklaşık 3,5 ve PC-3'e göre ise yaklaşık 4,2 kata kadar daha hızlıdır. Virtex 4'te ise, PC-1'e göre 4, PC-2'ye göre yaklaşık 3 ve PC-3'ye göre ise 3,6 kata kadar hız kazancı sağlanmıştır. Grafiklerde görülen hız kazançları tek bir çip üzerinde, tek bir modül konfigürasyonu çalıştırıldığında elde edilen sonuçlardır. Birden fazla çip üzerinde birden fazla modül yapılandırması paralel olarak çalıştırıldığında, bu hız kazançlarının katlanarak artacağı gayet açık bir şekilde görülebilmektedir.



Şekil 10. *Virtex5* FPGA çipine göre sentezlenmiş modülün PC'lerle başarım karşılaştırılması (The performance comparison of the module synthesized for *Virtex5* FPGA chip and the PCs)



Şekil 11. *Virtex4* FPGA çipine göre sentezlenmiş modülün PC'lerle başarım karşılaştırılması (The performance comparison of the module synthesized for *Virtex4* FPGA chip and the PCs)

4. SONUÇ VE TARTIŞMA (CONCULUSION AND DISCUSSION)

Görüntü filtreleme işlemi yazılımsal olarak yapıldığında çok fazla CPU zamanı gerektirmektedir. Bu çalışmada, görüntü filtreleme işlemini hızlandırmak amacıyla, FPGA çipleri üzerinde çalışabilecek donanım modülü tasarlanmıştır. Bu yaklaşımla, daha düşük maliyete sahip, daha yüksek performans gösteren ve çok kısa sürelerde defalarca programlanabilen daha esnek bir yapı ortaya konmuştur. Tasarlanan modül, gerçek veriler üzerinde işlemler yapılarak test edilmiş ve modülün ürettiği sonuçların doğrulanması yapılmıştır. Modülün farklı boyutlardaki görüntü verilerini işleme hızı değişik bilgisayarlarla karşılaştırılmıştır. Karşılaştırma sonuçlarına göre, tasarlanan modül kullanılarak görüntü filtreleme işleminin yaklaşık 2 ile 5 kat arasında hızlandırılabileceği tespit edilmiştir. Hız kazançları tek bir FPGA çipi üzerinde tek bir modülün çalıştırıldığı varsayılarak hesaplanmıştır. Bir FPGA çipine birden fazla modülün uygun şekilde yerleştirilmesiyle ya da birden fazla FPGA çipinde modülün birden fazla kopyasının çalıştırılmasıyla hız kazancının katlanarak artacağı düşünülmektedir. Söz konusu tasarım ASIC olarak gerçekleştirildiğinde ise, söz konusu hız kazançlarının yaklaşık iki katına çıkacağı düşünülmektedir.

5. KAYNAKLAR (REFERENCES)

[1]. Dehon, A., (2000). The Density Advantage of Reconfigurable Computing, *IEEE Computer*, 33, 41-49.

[2]. Qasim, S.M., Abbasi S.A., and Almashary, B., (2009). An Overview of Advanced FPGA Architectures for Optimized Hardware Realization of Computation Intensive Algorithms, *IMPACT'09*, 300-303.

[3]. Koyuncu, İ., (2008). A Matrix Multiplication Engine for Graphic Systems Designed for run on FPGA Devices, *Düzce Üniversitesi, Yüksek Lisans Tezi*, Düzce, Türkiye.

[4]. Kumar, P. , Suresh, S. , Perinbam, R. (2005). Digital Image Filter Design using Evolvable Hardvare", *4th Annual ACIS International Conference on Computer and Information Science*, 483-488.

[5]. Tang, K. and Astola, J., (1995). Nonlinear Multivariate Image Filtering Techniques", *IEEE Transactions on Image Processing*, 4(6), 788-798.
[6]. Kepkep, A., Gurkaynak, F.K., Ozdemir, H., Cilingiroglu, U., (1997). A Fully Pipelined Programmable Real-Time (3×3) Image Fitler Basedon Capacitive Threshold-Logic Gates, *IEEE Internetional Symposium on Circuits and Systems*, 3, 2072-2075.
[7]. Yano, Y., Hashiyama, T. and Okuma, S, (1999). On-line filter generation for binary image processing using FPGA, *IEEE Conference Proceedings on Systems, Man, and Cybernetics*, 5, 565-570.
[8]. Fu B., Xiong X., Sun G., (2011). An efficient mean filter algorithm, *IEEE/ICME International Conference on Complex Medical Engineering (CME)*, 466-470.