HDTVler için Düşük Karmaşıklıklı Gömülü Sıkıştırma
Low-Complexity Near-Lossless Embedded Compression for HDTVs

Okan Palaz[1,2], H. Fatih Uğurdağ[1,2], Buğra Kertmen[2], Özgür Özkurt[2], İsmail Gerçek[2], Faruk Dönmez[2]

[1] Özyeğin Üniversitesi, İstanbul, Türkiye
[2] VESTEK Araştırma Geliştirme A.Ş, İstanbul Türkiye

Özet

HDTV video işlemcileri çerçeve hızı değiştirme, binişme giderme ve diğer video iyileştirme işlemlerini yapabilmek için, bir ya da daha fazla geçmiş çerçeveyi hafızada tutmak durumundadır. HD çözünürlüklerde sıkıştırılmamış çerçeve okuyup yazma işlemi yüksek bant genişliği gerektirir. Bu yüksek bant genişliği çerçeveleri sıkıştırarak azaltılabilir. Video sıkıştırma yöntemleri bu amaçta kullanım için uygun değildir, çünkü bu yöntemler ağ trafiğini azaltırken fazladan hafiza trafiği yaratırlar. Bu amaca yönelik sıkıştırma yönteminin yüksek performanslı, düşük karmaşıklıklı ve kayıpsız (ya da kayıpsıza yakın) olması gerekir. Bu makalede, özgün uçtan uca bir gömülü sıkıştırma yöntemi önerilmektedir. Önerilen yöntem, 180nm ASIC üzerinde 4K Ultra HD çözünürlükte 30 Hz frekansında çalışmaya yeterli performansa sahip olup benzer çalışmaların 2 katına yakın saat frekansına erişebilmektedir.

Abstract

HDTV Video processors need to keep one or more uncompressed previous frames as they process streaming video when performing tasks such as frame rate conversion, deinterlacing and other video enhancement techniques. Reading and writing frames requires a lot of bandwidth at HD resolutions. This bandwidth can be reduced by applying compression. Video compression methods do not address this problem as they reduce network traffic while adding extra memory traffic. What is needed is high-performance, low-complexity, and lossless (or near-lossless) image compression. This paper presents a novel end-to-end embedded memory compression solution. It can support 4K Ultra HD video streams at 30 Hz on a 180nm ASIC, while clocking at rates close to twice of its competitors

1. Giriş

HDTV video işlemci tümdevrelerinde hafıza bant genişliği gereksinimleri yüksek seviyededir. Bunun sebebi bu sistemlerin girdisinin sıkıştırılmamış piksel verisi olmasıdır. HDTVlerde olması beklenen çerçeve hızı değiştirme, binişme giderme vb. gibi işlemler RAM'de tutulan çerçevelere erişebilmeyi gerektirir. Full-HD (1920x1080) çözünürlüğündeki sıkıştırılmamış bir çerçeve 48 Mbit boyutu kadardır ve 512 Mbit boyutunda bir RAM bile bu çerçevelerden birçok sayıda tutabilir. Ancak, çerçevelerin birden fazla kez yazılıp okunduğu durumda hafıza bant genişliği sınırlayıcı hale gelebilir. 60 Hz frekansında 1920x1080 çözünürlüklü bir akışın kullandığı bant genişliği 3 Gbps'ye yakındır. Yüksek bant genişliği gereksinimlerini karşılamak için RAM çipi yüksek saat frekanslarında sürülebilir ancak bu güç tüketim seviyesini yükseltir. Başka durumlarda bant genişliği ihtiyacını karşılamak için fazladan RAM çipi eklenebilir ancak bu maliyeti yükseltir.

Hafıza bant genişliğini azaltmak için piksel verisini RAM'e yazılmadan önce sıkıştıran Gömülü Sıkıştırma (GS) yöntemleri kullanılır. GS yöntemleri bazen video ve resim sıkıştırma tümdevrelerinde çerçeveleri saklamak ve okumak için ham piksel verisi yazmaya alternatif olarak kullanılmaktadır.

JPEG ve PNG gibi yöntemler bant genişliği sorununu çözmeye yönelik değildir, çünkü bunların kullanım amacı bant genişliğini değil toplam saklama alanını azaltmaktır. JPEG-LS [1] yönteminin düşük karmaşıklığı ve düşük hafıza kullanımı gibi bazı özellikleri bu senaryoya uymaktadır. Ancak, JPEG-LS'in bağlam modelleme sistemi bu yöntemi gerçek zamanlı uygulamalar için uyumsuz kılan bağımlılıklar oluşturmaktadır. [2], [3], ve [4] gibi JPEG-LS'in boruhatlı tasarımları önerilmiştir ancak bu çalışmalarda açıcı tasarımı bulunmamaktadır. [5], [6], [7], ve [8] çalışmalarında önerilen GS yöntemleri MPEG-2, H.264, ve JPEG2000 gibi video ve resim sıkıştırma tümdevrelerinde kullanılmak üzere geliştirildiklerinden HDTV senaryosu gibi satır satır tarama ile uyumlu değillerdir. [9] çalışmasında önerilen yöntem satır satır tarama senaryosuna uygundur, ancak gerekli performansı elde edebilmek için birden fazla çekirdeğe ihtiyaç duymuşlardır.

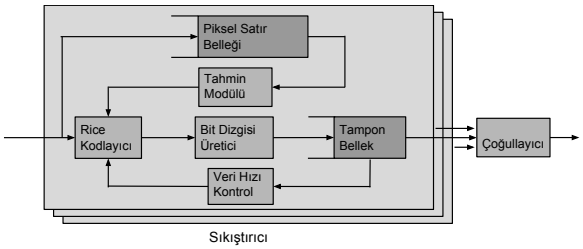
Bu çalışmada, bant genişliğini sabit Hedeflenen Sıkıştırma Oranına (HSO) garantili sıkıştıran, satır satır tarama senaryosunda kullanım için tasarlanmış küçük kaplama alanlı bir sıkıştırma yöntemi ve tasarımı önerilmektedir. Çalışmanın ana hedeflerinden biri 1920x1080 çözünürlüğünü 60 Hz frekansında işlemek için gerekli çıkan iş oranını Xilinx Spartan-6 FPGA'inde yakalamaktır.

2. Önerilen Yöntem

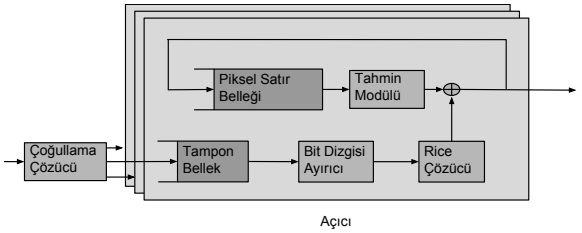
Bu bölümde algoritma ve tasarımı detaylı olarak açıklanmıştır.

2.1. Genel Bakış

Tasarımın en üst katmanında 2 tane alt modül bulunmaktadır; sıkıştırıcı ve açıcı modülleri. Sonraki bölümlerde bu modüllerin çalışma yapılarından bahsedilmektedir.



Şekil 1: Sıkıştırıcının yapısı.



Şekil 2: Açıcının yapısı.

2.1.1. Sıkıştırıcı

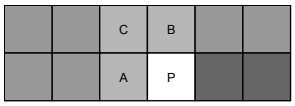
Sıkıştırıcı Şekil 1'de gösterilen modüllerden oluşmaktadır. Sıkıştırma algoritması tahmin ve artık kodlama ile yapılmaktadır. Her piksel için bir değer tahmin edilir ve gerçek değer ile tahmin arasındaki fark Golomb-Rice yöntemi ile kodlanır. Tahmin etme işleminde kullanmak üzere 1 satırlık piksel verisi hafızada saklanır. Farklı uzunluktaki Golomb-Rice kodları bit dizgisi üretici modülünde art arda eklenir ve 32 bit boyutunda kelimelere ayrılır. Bu 32 bit boyutundaki kelimeler tampon belleğe yazılır. Bu tampon belleğin taşmamasını ve tamamen boşalmamasını veri hızı kontrol modülü önler. 3 renk kanalını sıkıştırmak için sıkıştırıcı modülünden 3 tane kullanılmaktadır. Çoğullama modülü 3 renk kanalının tampon belleklerinden 32 bit boyutundaki kelimeleri sırasıyla alıp RAM'e yazar.

2.1.2. Açıcı

Şekil 2'de gösterilen açıcı modülü sıkıştırıcının simetriğine yakın bir yapıya sahiptir. Sıkıştırıcıdaki gibi 3 kanal için 3 açıcı kullanılmaktadır. Çoğullama çözücü 32 bit boyutundaki kelimeleri RAM'den okuyup, bu kelimeleri o kelimeye denk gelen renk kanalının tampon belleğine yazar. 32 bit boyutundaki kelimeler bit dizgisi ayırıcı modülü tarafından alınıp, Golomb-Rice sembollerine bölünür. Golomb-Rice çözücü modülü artık değerini çözer. Artık değeri tahmin edilen piksel değerine eklenilir ve pikselin gerçek değeri hesaplanır. Hesaplanan piksel değeri piksel satır belleğinde ileride tahmin için kullanılmak üzere saklanır.

2.2. Nicemleme

Önerilen sıkıştırma algoritması tampon belleğin dolmasını engellemek için nicemleme kullanır. Kullanılan yöntemde pikseller 2'nin kuvvetleri ile nicemlenir. Bu işlem ikili sistemde en az anlamlı bitleri kırpmak ile eşdeğerdir. Bu yüzden basit bir kaydırma devresi ile gerçeklenebilir. Kullanılan nicemleme seviye



Şekil 3: Tahmin için kullanılan komşu pikseller.

değeri n, kırpılan bit sayısına denk gelmektedir. Pikseller yeniden oluşturulurken, kırpılan bitler her zaman 8 bitlik bir değer olan tahmin edilen piksel değerinden kopyalanarak tamamlanır. Nicemleme seviyesi veri hızı kontrol algoritması tarafından belirlenir. Bu algoritmanın detayları Bölüm 2.5'te verilmiştir.

2.3. Tahmin Yöntemi Çevrimi

Piksel tahmin algoritması birden fazla tahmin algoritmasını kullanır. Tahmin algoritmaları arasından yapılan seçim, açıcı tarafından tekrarlanılabilecek şekilde bit dizgisine fazladan veri girilmeden yapılır. Periyodik kontrol noktalarında, önceki kontrol noktasından itibaren bit dizgisine yazılan bit sayısı önceden belirlenmiş eşik değeri ile karşılaştırılır. Eğer ki yazılan bit sayısı bu eşik değerini geçti ise, tahmin yöntemi listede bir sonra gelen yönteme ayarlanır ve sonraki kontrol noktasına kadar bu yöntem kullanılır. Kullanılan her tahmin yöntemi için eşik değerleri farklı ayarlanarak belirli yöntemlere öncelik verilmesi sağlanabilir. İşlenilen çerçevelerin farklı bölgeleri farklı karakteristikler gösterebileceğinden bu yöntem sıkıştırma verimini yükseltir. Örnek olarak bir çerçevede bir bölge dikey dokulardan oluşuyor iken, aynı çerçevede başka bir bölge yatay dokulardan oluşuyor olabilir. Böyle bir senaryoda tahmin etme yöntemi Şekil 3'te gösterilen A, B ve C komşu pikselleri arasında dönebilir. Yapılan gerçeklemede bu şekilde 2 farklı piksel tahmin yöntemi kullanılmıştır. JPEG-LS'in tahmin yöntemi ile C pikselini tahmin etme yöntemi.

JPEG-LS tahmin yöntemi düşük karmaşıklığı ve doğal görsellerdeki performansından dolayı önerilen sistem için uygun bir tercihtir. Bu yöntemde 3 komşu piksel A, B ve C kullanılır. Kullanılan komşu pikseller arasında dikey uzaklığı en fazla olan piksel 1 uzaklığında olduğundan, satır satır tarama uygulamasında yalnızca bir satırlık geçmiş piksel verisinin tampon bellekte tutulması yeterlidir. JPEG-LS tahmin yöntemi bir kenar sezicisi gibi çalışır. Eğer ki sezilen kenar yatay ise A pikseli tahmin edilir, sezilen kenar dikey ise B pikseli tahmin edilir. Diğer durumlarda A + B - C değeri tahmin edilir. Bilgisayar çıktısı gibi sentetik görsellerde, dama tahtası gibi dokular olabilir. Bu durumda JPEG-LS'in tahmin yöntemi başarısız olur iken C pikselini tahmin etmek başarılı sonuç vermektedir.

2.4. Golomb-Rice Kodlama Yöntemi

Artık değerini kodlamak için Golomb-Rice kodlama yöntemi kullanılır. Golomb-Rice kodlama yöntemi sabit geometrik dağılım varsaysan bir entropi kodlama yöntemidir. Küçük değerler az sayıda bit ile kodlanırken, büyük değerler daha çok bit kullanılarak kodlanılır. Golomb-Rice kodları iki parçadan oluşur. Kodlanacak pozitif tam sayı değeri ikinin kuvveti olan bir sayı ile bölünür. Bölüm değeri tekil kodlama yöntemi ile yazılır. Kalan değeri ise ikili sistemdeki değeri ile sabit boyutlu kısımda yazılır. İki kısım "0" bitlerinden sonra gelen ilk "1" değerindeki dur biti ile ayrılır. Artık değeri pozitif ya da negatif olabilece-

Şekil 4: Golomb-Rice kodlama sözdizim yapısı.

ğinden, sembolün sonunda işareti belirten bir işaret biti koyulur. Sembollerde ikili sistemde yazılan parçanın boyutu sabittir. Yalnızca tekil kodlama yöntemi ile kodlanılan kısım değişken uzunluktadır. Bit dizgisi okunurken sembolün uzunluğunu belirlemek yalnızca dur biti olan "1" bitinin bit dizgisindeki yerini bulmak yeterlidir. Bundan dolayı sembol uzunluk hesaplama işlemi oldukça kısadır. Kullanılan sözdizim yapısı Şekil 4'te gösterilmiştir. Maksimum tekil kod uzunluğunu belirten DURMA değeri 10 olarak tanımlanmıştır. Artık değerinin büyük olduğu durumlarda Golomb-Rice kodlaması yerine DURMA kadar "0" biti yazılır ve ardından pikselin gerçek değeri yazılır.

2.5. Veri Hızı Kontrolü

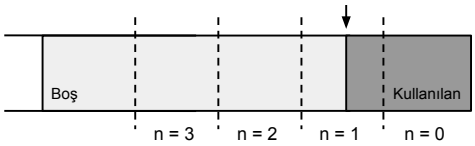
Veri hızı kontrol sistemi tampon belleğe giren verinin miktarını ayarlamak ile görevlidir. Tampon belleğin boşalma hızı sabittir ve HSO ile belirlenmiştir. Veri hızı kontrol sistemi tampon belleğin hiç bir zaman tamamen dolmamasını ve tamamen boşalmamasını sağlamak zorundadır.

Tampon bellekten veri okunduğu her çevrimde tampon bellekte anlamlı veri olması gerekmektedir. Çünkü üç renk kanalından alınan kelimeler çoğullayıcı tarafından belirli bir sıraya getirilip RAM'e yazılmaktadır. Kelimelerin okunacağı çevrimler bellidir ve bekletilemez. Tampon bellek boş iken yapılacak olan okuma işleminde anlamsız veri alınacaktır.

Tamamen boşalma kontrolü, veri hızı kontrol modülü tarafından kaldırılan bir bayrak ile sağlanır. Eğer tampon bellek önceden belirlenmiş bir eşik değerine kadar boşaltılır ise BOŞAYAKIN bayrağı "1" değerine ayarlanır. BOŞAYAKIN bayrağı "1" olduğu durumda Golomb-Rice kodlama modülü pikselleri, artık değeri yüksek bir değermiş gibi kodlar. Bu durumda oluşturulan sembollerin boyutu DURMA + 8 bit olur ve tampon bellek boşaltıldığı hızdan daha hızlı doldurulmuş olur. Böylece tampon bellek hiç bir zaman tamamen boşaltılmamış olur.

Taşma kontrolü kayıplı moda geçilerek, yani nicemleme yapılarak, sağlanır. İşlenilen çerçeve yan yana gelen sabit boyuttaki piksel bloklarına ayrılacak şekilde bölünür. Her bloğun ilk pikselinden önce tampon belleğin durumuna göre nicemleme seviyesi belirlenir ve bloğun sonuna kadar bu nicemleme seviyesi kullanılır. Nicemleme seviyesi, tampon belleğin durumundan hesaplandığı için bit dizgisine yazılması gerekmez.

Taşma ve tamamen boşalma durumlarını engellemek, veri hızı kontrol sisteminin tek amacı değildir. Veri hızı kontrol algoritması yan yana olan iki bloğun nicemleme seviyelerinin artışı 1'i geçmeyecek şekilde ayarlanmıştır. Böylece nicemleme seviyesi değişimi akıcı bir şekilde olur ve ani değişikliklerden oluşacak gözle görülebilecek bozukluklar engellenmiş olur. Veri hızı kontrol algoritması nicemleme seviyesini belirlemek için tampon belleğin durumuna bakar. Her nicemleme seviyesi ihtimali için Şekil 5'te görüldüğü gibi bir eşik değeri tanımlanmıştır. Eşik değerleri arasındaki fark, bir blok boyunca olabilecek maksimum oynama miktarına göre hesaplanmıştır. Bu değerler seçilen HSO için sabit olduğundan, bu eşik seviyeleri derleme süresinde tanımlanır.



Şekil 5: Nicemleme seviyesinin eşik değerlerine göre belirlenmesi.

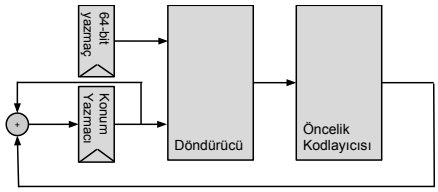
Şekil 6: Aynı piksel işlenirken sıkıştırıcı ve açıcıdaki tampon belleklerin simetri durumu.

Veri hızı kontrol sistemi, simetri özelliği ile açıcı modülü için de bu kontrolü yapmış olur. Simteri, sıkıştırıcı ve açıcının tampon belleklerinin doluluklarının birbirinin zıttı olmasını sağlar. Bir piksel sıkıştırılırken tampon belleğe yazılan bit sayısı, açma işleminde tampon bellekten okunan bit sayısına eşit olacaktır. Ayrıca sıkıştırıcıda tampon belleğin boşlatılma hızı, açıcıdaki tampon belleğin doldurulma hızına eşit olacağından tampon bellek durumları Şekil 6'da gösterildiği gibi birbirinin simetriği olacaktır. Bu simetrik davranıştan ötürü, açıcı nicemleme seviyesini de tampon belleğin durumundan okuyabilir. Bu yüzden nicemleme seviyesinin değerini bit dizgisine yazmaya gerek yoktur.

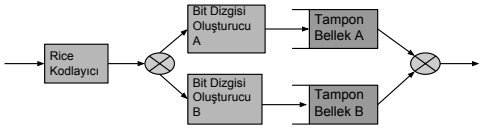
2.6. Birden Fazla Bit Dizgisi Oluşturulması

Devrenin ulaşabileceği minimum saat periyodunu belirleyen devre parçası, Şekil 7'de gösterilen açıcıdaki bit dizgisi ayırıcı modülündedir. Birbiri ardına eklenmiş farklı uzunluktaki Golomb-Rice sembolleri 32 bit boyutundaki kelimeler halinde tampon bellekten okunur. Golomb-Rice sembollerinin başlangıçları 32 bit boyutundaki kelimelerin belirli bir noktasına denk gelmez ve 2 farklı 32 bit parça arasına bile düşebilir. Herhangi bir çevrimde sembolün tamamına erişebiliyor olmak için 2 tane kelime okunmuş ve 64 bit boyutunda bir yazmaçta tutulmuş halde bekletilir. 64 bitlik döndürücü kullanılarak, yazmaçta herhangi bir konumda olabilecek sembol, başlangıç bitinden başlayacak şekilde Golomb-Rice çözücüye beslenir. Konum değeri bir yazmaçta tutulur ve sonraki çevrime hazır olacak şekilde güncellenmesi gerekir. Bu konum değerinin oynama miktarı ancak çözümlenme işleminden sonra belirlenebilir. Sembolün uzunluğunu hesaplamak için yalnızca ilk "1" bitinin bulunması gerekmektedir. Yalnızca bir tane uzunluk hesabı olmasına rağmen kritik yol uzun çıkmaktadır ve performans hedefine ulaşılamamıştır.

Gerekli performansa ulaşabilmek için çift bit dizgisi çözümü getirilmiştir. Bu yaklaşım ile konum yazmacını güncelleme işlemi iki çevrime bölünmüştür. Bunu gerçeklemek için, Şekil 8'de gösterildiği gibi bit dizgisi oluşturucunun ve tampon belleklerin tekrarlanması gerekir. Her çevrimde tampon bel-



Şekil 7: Minimum saat periyodunu belirleyen kritik yol.



Şekil 8: Kullanılan çift bit dizgisi oluşturma yöntemi.

lek gösterici güncellenir ve yaratılan sembol gösterilmekte olan tampon belleğe yazılır. Bu kelimeler sırasıyla A ve B tampon belleklerinden alınıp RAM'e tek bir bit dizgisi halinde yazılır.

2.7. Çoğullama

Sıkıştırıcıdan çıkan verinin bant genişliği bir jeton kova algoritması ile sağlanır. Hedef bit hızı, 4 biti virgül öncesi ve 4 biti virgül sonrası olacak şekilde 8 bitlik bir sabit virgüllü veri değeri ile belirtilir. Her çevrimde, bir sayaç bu bit hızı değeri kadar arttırılır. Sayacın değeri kelime genişliği olan 32'yi geçtiği çevrimde tampon bellekten 1 kelime okunur ve sayaç 32 azaltılır.

3 kanalın verisinin aynı oranda sıkıştırıldığı durumda 3 kanalın kelimelerinin sıralanması işlemi basittir. Ancak, YCbCr renk uzayında çalışıldığında bu yaklaşım verimli olmaz. Çünkü insan gözü Y kanalına daha hassastır ve neredeyse tüm video kaynakları Y kanalının diğer kanallara göre 4 kat daha fazla örneklendiği YUV4:2:0 formatındadır. Bu yüzden Y kanalına daha fazla bant genişliği vermek algılanan görsel kalite seviyesini arttırır. Bunu sağlayabilmek için öncelikli kuyruk kullanan çoğullayıcı kullanılmıştır. Bu yapıda üç jeton kova yapısı paralelde çalışır. Kelimesi okunmaya hazır olan kanal bunu belirten bir bayrak kaldırır ve sıradaki çevrimde bu kanalın tampon belleğinden bir kelime okunup RAM'e yazılır. Birden fazla jeton kovası aynı çevrimde bayrak kaldırdığı zaman paralelde hepsi birden okunur; ancak RAM'e yazılırken öncelik sırasıyla yazılır. Önceliği az olan kanalların kelimeleri, yüksek öncelikli kelimeler yazılana kadar bir yazmaçta bekletilir. Bu algoritma belirlenimci bir kelime sıralaması oluşturacağından açıcı tarafından da tekrar edilebilir. Bu yüzden kelimeleri bir araya getirirken kanal numarası ile işaretlemeye gerek kalmamaktadır.

3. Sentez Sonuçları ve Performans Analizi

Bu bölüm sentez ve sıkıştırma performansı test sonuçlarını listelemektedir. Elde edilen sonuçlar [9] çalışması ile karşılaştırılmıştır.

3.1. Sentez Sonuçları

Önerilen tasarım hem FPGA hem ASIC sentez programları ile sentezlenmiştir. Bu sonuçlar Tablo 1'de gösterilmiştir.

FPGA sentez sonuçları hedef FPGA üzerindeki kaynakların yalnızca %15'i kadarının kullanıldığını göstermektedir. Erişilen saat frekansında elde edilen çıkan iş miktarı 1920x1080 çözünürlüğünü 60 Hz frekansında işlemeye yeterli seviyededir.

ASIC sentez sonuçları [9] çalışmasından alınan sonuçlar ile karşılaştırılmıştır. Bu sonuçlar göstermektedir ki, benzer konfigürasyonlar kullanıldığında önerilen tasarımın ulaşabildiği saat frekansı diğer çalışmada ulaşılmış değerin %195'i kadardır. Ortaya çıkan iş miktarı birden fazla çekirdeğe ihtiyaç duymadan aynı seviyeye yaklaşmaktadır. Ayrıca önerilen tasarımın kullandığı mantık geçidi sayısı daha azdır.

Tablo 1: FPGA ve ASIC Sentez Sonuçları

FPGA Sentez			
Hedef FPGA Çipi	Xilinx Spartan-6 XC6SLX45		
Sentez Programı	Xilinx ISE 14.6		
Maks. Saat Frek.	154 MHz		
LUT Kullanımı	4272/27288 (15%)		
Blok RAM Kullanımı	14/116 (12%)		
ASIC Sentez			
	Önerilen	Önerilen	[9]
Stdcell Kütüphanesi	TSMC 180nm Slow	TSMC 180nm Typical	TSMC 180nm Typical
Wire Load	wl10	wl10	wl10
Maks. Saat Frek.	235 MHz	391 MHz	200 MHz
Mantık Geçit Sayısı	36 Bin	35 Bin	45 Bin

3.2. Sıkıştırma Performansı

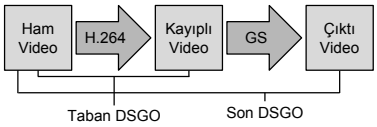
Algoritmanın performansını değerlendirmek için 3 farklı HSO değerinde testler yapılmıştır. Testlerde algoritmanın performansı Doruk Sinyal Gürültü Oranı Kaybı (DSGO Kaybı), yani uygulanan GS yönteminin var olan kaybın üzerine eklediği kayıp miktarıyla ölçülmüştür. DSGO Kaybı değeri, Şekil 9'da gösterilen Taban DSGO değeri ile Son DSGO arasındaki fark alınarak hesaplanır.

Yapılan testler için kayıpsız video kaynakları H.264 ile önceden 3 farklı Nicemleme Parametre (NP) değerleri 15, 20 ve 25 ile sıkıştırılmıştır. Bu testlerin sonuçları Tablo 2'de gösterilmiştir. Tabloda gösterilen DSGO değeri dB Taban DSGO değeridir. Gri renkli sütunlarda gösterilen DSGO Kaybı değerleri [9] çalışmasından alınan değerler ile karışlaştırılmıştır.

Önerilen algoritma daha yüksek saat frekansına ulaşabilmesine ve daha düşük karmaşıklık seviyesinde olmasına rağmen, sonuçlar elde edilen görsel kalitenin karşılaştırılabilir seviyelerde olduğunu göstermektedir. [9] çalışması ile ortalama değerleri karşılaştırıldığında, HSO 2,0 ve HSO 2,5 seviyelerinde önerilen algoritma daha iyi performans vermektedir. Ancak, HSO 3,0 iken yalnızca NP 15 seviyesinde daha iyi performans vermiştir.

4. Gelecek Çalışmalar

Yapılan gerçeklenim ile erişilen iş miktarı çıktısı gerekli hedefi tutturmuştur. Ancak, çevrim başına alınan iş oranı yalnızca 1



Şekil 9: DSGO Kaybı metriğinin hesaplanmasında kullanılan DSGO değerleri.

Tablo 2: H.264 Video ile DSGO Kaybı Test Sonuçları

	HSO	2,0		2,5		3,0	
Kaynak	DSGO		[9]		[9]		[9]
Bluesky@NP15	48,06	0,01	0,01	0,15	0,24	3,38	5,17
Rush@NP15	48,41	0,00	0,01	0,00	0,28	1,83	5,56
Station2@NP15	47,25	0,00	0,01	0,01	2,78	2,55	9,08
Sunflower@NP15	47,26	0,00	0,01	0,00	0,11	1,05	2,49
Bluesky@NP20	44,35	0,00	0,01	0,05	0,01	1,53	0,08
Rush@NP20	45,19	0,00	0,01	0,00	0,01	0,23	0,01
Station2@NP20	43,87	0,00	0,00	0,00	0,04	0,80	2,72
Sunflower@NP20	44,89	0,00	0,01	0,00	0,01	0,37	0,37
Bluesky@NP25	41,21	0,00	0,01	0,02	0,01	0,69	0,28
Rush@NP25	44,14	0,00	0,01	0,00	0,01	0,03	0,01
Station2@NP25	42,40	0,00	0,01	0,00	0,01	0,37	0,38
Sunflower@NP25	43,10	0,00	0,01	0,00	0,01	0,18	0,10

piksel kadardır. Benzer çalışmalardaki gibi birden fazla sıkıştırıcı ve açıcı çekirdekleri kullanılarak iş oranı 2 ya da daha fazla katına arttırılabilir. 2 katına artma durumunda aynı FPGA üzerinde 4K Ultra HD çözünürlük (3840x2160) 30 Hz frekansta işlenilebilir. Paralel çalışmayı desteklemek için çekirdeklerin işlediği piksellerin sıkıştırılma ve açma işlemleri için birbirine bağlı olmamaları gerekir. Var olan tasarıma az miktarda eklemeler yapılarak bu davranış sağlanabilir.

Rasgele erişim sağlamak başka bir geliştirme imkanı olabilir. Sabit HSO özelliği kullanılarak, satırlar RAM'de belirli adreslere yazılabilir ve satır tabanlı rasgele erişim sağlanabilir.

5. Sonuç

Bu makalede; satır satır tarama girdi senaryosu için FPGA ve ASIC uyumlu, neredeyse kayıpsız bir GS algoritması önerilmiştir. Önerilen algoritma, çoğu video kaynağını HSO 2,0 seviyesinde kayıpsız sıkıştırabilmekte ve 2,5 oranında da neredeyse kayıpsız seviyesini korumaktadır. Yapılan tasarımın Xilinx Spartan-6 FPGA üzerindeki gerçeklemesi, Full-HD (1920x1080) çözünürlükte çerçeveleri 60 Hz frekansında sıkıştırabilmektedir. 180 nm ASIC için gerçeklemesi ise 4K Ultra HD (3840x2160) çözünürlükte çerçeveleri 30 Hz frekansında sıkıştırabilmektedir. Benzer çalışmalar ile karşılaştırıldığında algoritmanın sıkıştırma performansı yakın seviyelerde olmasına rağmen, ulaşılan saat frekansı daha yüksek ve kaplama alanı daha küçük çıkmıştır.

6. Kaynaklar

[1] M. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, 2000.

[2] X. Li, X. Chen, X. Xie, G. Li, L. Zhang, C. Zhang, and Z. Wang, "A low power, fully pipelined JPEG-LS encoder for lossless image compression," in *IEEE International Conference on Multimedia and Expo*, 2007.

[3] M. Papadonikolakis, V. Pantazis, and A. P. Kakarountas, "Efficient high-performance ASIC implementation of JPEG-LS encoder," in *Design, Automation & Test in Europe Conference & Exhibition*, 2007.

[4] H. Daryanavard, O. Abbasi, and R. Talebi, "FPGA implementation of JPEG-LS compression algorithm for real time applications," in *Iranian Conference on Electrical Engineering*, 2011.

[5] J. Kim and C.-M. Kyung, "A lossless embedded compression using significant bit truncation for HD video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 6, pp. 848–860, 2010.

[6] C.-H. Son, J.-W. Kim, S.-G. Song, S.-M. Park, and Y.-M. Kim, "Low complexity embedded compression algorithm for reduction of memory size and bandwidth requirements in the JPEG2000 encoder," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2421–2429, 2010.

[7] W.-Y. Chen, L.-F. Ding, P.-K. Tsung, and L.-G. Chen, "Architecture design of high performance embedded compression for high definition video coding," in *IEEE International Conference on Multimedia and Expo*, 2008.

[8] T. Y. Lee, "A new frame-recompression algorithm and its hardware design for MPEG-2 video decoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 6, pp. 529–534, 2003.

[9] T.-H. Tsai and Y.-H. Lee, "A 6.4 Gbit/s embedded compression codec for memory-efficient applications on advanced-HD specification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 10, pp. 1277–1291, 2010.