9. ULUSAL HAVACILIK VE UZAY KONFERANSI
14-16 Eylül 2022, İzmir Ekonomi Üniversitesi, İzmir

UHUK-2022-151

6 SERBESTLİK DERECESİNE SAHİP BİR SİSTEM MATEMATİK MODELİNDE DÖNÜŞÜM MATRİSİNİN FPGA ÜZERİNDE DONANIM TABANLI GERÇEKLENMESİ

Sezer Memiş*, Ramazan Yeniçeri†
İstanbul Teknik Üniversitesi, İstanbul

ÖZET

Bu bildiride, 6 serbestlik derecesine sahip bir çok rotorlu matematik modelinin eksen dönüşüm matrisinin FPGA üzerinde donanım tabanlı gerçeklemesi çalışılmıştır. Çalışmada, gerçek zamanlılık ile model temsiliyet seviyesi arasında ödünleşme sorununa sahip yazılım tabanlı gerçeklemeye alternatif olarak donanım tabanlı gerçekleme yöntemi önerilmektedir. Bu yöntem, model doğruluğundan ödün vermeden simülasyon hızını artırırken hava aracı tasarım maliyetlerini azaltmaktadır. Entegrasyon işleminde model güvenilirliğinin ve modülerliğin sağlanabilmesi için model tabanlı tasarım ve otomatik kod üretme yaklaşımları kullanılmıştır. FPGA kaynak kullanımının azaltılması amacıyla standart işlemci mimarisinde tasarlanan bloklarda kaynak paylaşımı yaklaşımı uygulanmıştır.

GİRİŞ

Dinamik sistemlerin matematik modellerinin elde edilmesi ve elde edilen bu modeller kullanılarak problem isterleri ile maliyet ve kaynak kısıtlarına uygun tasarımların yapılması, diğer tüm mühendislik disiplinlerinde olduğu gibi havacılık ve uzay teknolojileri geliştirirken de uygulanması kaçınılmaz bir yöntemdir.

Elde edilen matematik modeller, üzerinde çalışılan sistemin dinamik davranışının analiz edilmesi için kullanılır. Yapılan bu analizler daha ileri çalışmalar (örneğin kontrolcü tasarımı) için temel ihtiyaç konumundadır. Matematik modeller tasarımcıya sistemi manipüle etme imkânı vermektedir [Zipfel, 2007].

Sistemlerin gerek doğal gerekse manipüle edilmiş halinin dinamik davranışı belirli çıktılar üretmektedir. Bu çıktıların, tasarım süresi başından kullanıcıya sunulana kadar sürekli olarak elde edilip sistemin test edilmesi gerekmektedir. Bu test işlemleri günümüzde dijital ortamda simülasyonlar yapılarak gerçekleştirilmektedir.

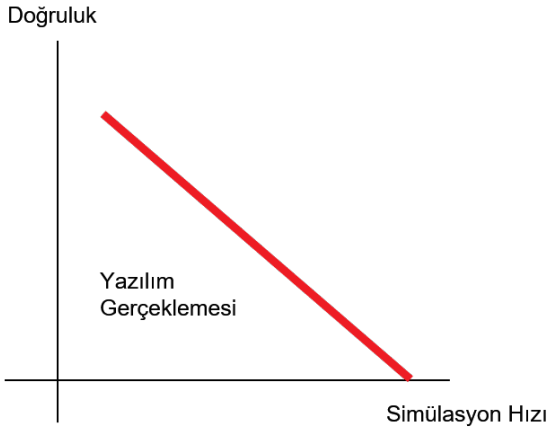
*Araştırma Görevlisi, Savunma Tek. Böl., E-posta: memis20@itu.edu.tr
†Dr. Öğr. Üyesi, Uçak Müh. Böl. E-posta: yenicerir@itu.edu.tr

Simülasyonlar, gerçeklendiği dijital ortamın türüne göre donanım veya yazılım tabanlı olarak uygulanabilir. Yazılım tabanlı simülasyon yöntemi ilgili programlama dillerinin kolay öğrenilebilirliği ve uygulanabilirliği sayesinde günümüzde yaygın olarak kullanılmaktadır.

Günümüzde çeşitli faaliyet alanlarında kullanılmak üzere insansız hava araçlarına olan talebin yükselmesiyle hem sektörde hem de akademide bu alandaki araştırma geliştirme faaliyetleri oldukça artmıştır. Modelleme, kontrolcü/kestirimci tasarlama, prototipleme ve uçuş testleri İHA geliştirme süreçleri için hızın yüksek öneme sahip olabileceği adımlardır.

Yazılım tabanlı simülasyon yöntemine dayalı uygulamalar simülasyonun koşma süresini kısaltırken fiziksel sistemin, örneğin çok rotorlu, dinamiğinin temsil seviyesinden ödün verilmesine neden olmaktadır. Daha açık bir ifadeyle, fiziksel sistemi temsil ettiği kabul edilen matematik model ne kadar sadeyse yazılım tabanlı simülasyon o kadar hızlı çalışır. Tam tersi durumda da matematik model ne kadar çok fiziksel fenomeni temsil ediyorsa yazılım tabanlı simülasyon o kadar uzun sürmektedir. Matematik model sadeliği eğilimindeki bu ödünleşme havacılık uygulamalarında kritik denilebilecek tasarım eksikliklerine sebep olabilmektedir. Örneğin bir İHA için açı kontrolcüsü tasarlanırken matematik modelde rüzgâr etkisinin eksikliği, uygulamada İHA'nın kararsız davranmasına ya da kontrolcünün fazla efor sarf etmesi sebebiyle bataryanın daha hızlı tükenmesine sebep olabilir.

Bir başka bakış açısıyla, havacılıkta gerçek zamanlı simülasyon yapabilmek çok kritik bir yetenektir [Bélanger, Venne ve Paquin, 2010]. Gerçek zamanlı simülasyonu yüksek temsiliyet seviyesiyle(doğrulukla) yapabilmek ise uçuş güvenliğini artırdığı ve tasarım süreçlerini kolaylaştırdığı için havacılıkta vazgeçilmez bir yöntem olmaktadır. Yazılım tabanlı simülasyonlar model doğruluğu arttıkça gerçek zamanlılık özelliğini kaybettiğinden havacılıkta uygulanması zayıf kalabilmektedir. Şekil 1'de yazılım tabanlı gerçeklemeyle simülasyon hızı arasındaki ilişki verilmiştir.



Şekil 1: Yazılım tabanlı simülasyonda model doğruluğu ile simülasyon hızı arasındaki ilişki

Daha önce de bahsedildiği üzere havacılıkta uçuş testleri, hıza ihtiyaç duyulan tasarım aşamalarından biridir. Burada maliyet, efor ve test sürelerinin uzunluğu sorunları ortaya çıkmaktadır. Ele alınan tüm uçuş senaryolarının test edilmesi uçuş güvenliği açısından elzemken hem denenebilecek senaryoların kısıtlı olması hem maliyetin oldukça artması hem de tüm bu sürecin çok uzun sürmesi havacılık uygulamaları için büyük bir problemdir.

Yukarıda bahsedilen; gerçek zamanlı simülasyon, doğruluk ve uçuş test problemlerine çözüm olarak Alanda Programlanabilir Kapı Dizisi (FPGA) üzerinde donanım tabanlı simülasyon yöntemi önerilebilir. Paralel işlem kapasiteleri, gecikmelerin temel lojik elemanların gecikmeleri mertebesinde oluşu FPGA'leri gerçek zamanlı simülasyonlar için oldukça uygun kılmaktadır [Bélanger, Venne ve Paquin, 2010]. Donanım tabanlı gerçeklemede simülasyon hızı yazılım tabanlıya göre oldukça yüksek seviyelere çıkabildiği için, yazılım tabanlı simülasyonlarda model doğruluğundan verilen

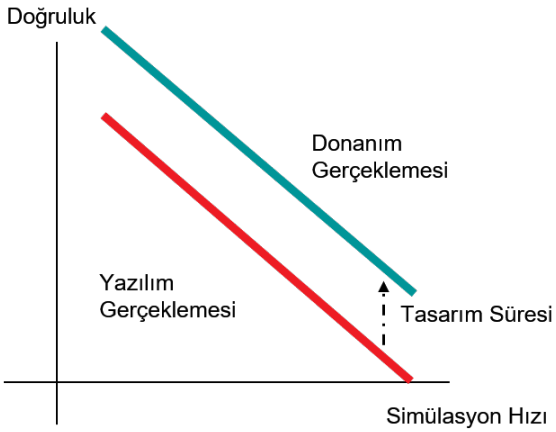
ödünün burada verilmesine gerek yoktur. Üzerinde çalışılan sistem detay fiziksel fenomenleriyle birlikte yüksek doğrulukla modellenebilir. Şekil 2'de donanım ve yazılım tabanlı gerçekleme kıyaslanmıştır.

Donanım tabanlı modelleme uçuş testleri yerine geçmesi de mümkündür. Tekrar tekrar çalıştırılabilen yüksek doğrulukla modellenmiş tek bir donanım tabanlı gerçeklemeyle pek çok uçuş test senaryosu simülasyon ortamında uygulanabilir. Bu sayede insan eforu ve test maliyetleri azaltılabilir. Bunun yanında zamandan fazlaca tasarruf edilmiş olunur.

FPGA'lar havacılık uygulamalarında kendine geniş yer bulmaktadır [Bouhali, Shamani, Dahmane, Belaidi ve Nurmi, 2017]. Otopilot [Konomura ve Koichi, 2014] ve model öngörülü kontrolcü [de Farias, Murilo ve Lopes, 2021] donanımı olarak kullanımları bu uygulamalara örnek olarak gösterilebilir. FPGA'ların, havacılıkta, matematik model simülasyon platformu olarak kullanılması bu çalışmanın katkıları arasında görülmektedir.

Havacılıkta uçak, roket, helikopter, çok rotorlu gibi insanlı/insansız hava araçlarıyla ilgili mühendislik uygulamalarında bu araçlar 6 serbestlik derecesine (6 DoF) sahip olacak şekilde modellenmektedir. 6 serbestlik derecesinde hareketi temsil eden denklemler öteleme ve dönme aksiyonlarından oluşmaktadır. Öteleme ve dönme hareketinin ayrı ayrı kinetik ve kinematik denklemleri toplam 6 DoF matematik modeli oluşturur. Burada kinetik denklemler Newton'ın hareket kanunlarından türetilirken kinematik denklemler eksenler arası dönüşüm ve integrasyon işlemlerinden oluşur.

Bu çalışmada, bahsedilen kinematik denklemler içerisinde eksenler arası dönüşüm işlemini yapan, trigonometrik ifadeler içeren dönüşüm matrisinin FPGA üzerinde donanım tabanlı gerçeklenmesi çalışılmıştır.



Şekil 2: Yazılım ve donanım tabanlı simülasyon hızlarıyla doğruluk seviyelerinin karşılaştırılması

DÖRT ROTORLU DİNAMİK MODELİ

6 serbestlik derecesine sahip bir İHA'nın dinamik davranışını temsil eden matematik modeli; öteleme kinetik (Denklem 1), kinematik (Denklem 2) ve dönme kinetik (Denklem 3), kinematik (Denklem 4) denklemleri olmak üzere 4 adet denklem sisteminden oluşmaktadır. Newton-Euler yöntemi kullanılarak elde edilen bu denklemler aşağıda verilmiştir.

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \begin{pmatrix} -g\sin\theta \\ g\cos\theta\sin\phi \\ g\cos\theta\cos\phi \end{pmatrix} + \frac{1}{m}\begin{pmatrix} 0 \\ 0 \\ -f \end{pmatrix} \tag{1}$$

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{2}$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{I_y - I_z}{J_x} qr \\ \frac{I_z - I_x}{J_y} pr \\ \frac{I_x - I_y}{I_z} pq \end{pmatrix} + \begin{pmatrix} \frac{1}{I_x}\tau_\phi \\ \frac{1}{I_y}\tau_\theta \\ \frac{1}{I_z}\tau_\psi \end{pmatrix} \tag{3}$$

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \psi \end{pmatrix} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \tag{4}$$

Denklem 2 ve 4'te verilen 3x3'lük matrisler dönüşüm matrisleri olarak adlandırılmaktadır. Denklem 2'deki dönüşüm matrisi, cisim eksenindeki çizgisel hızların atalet eksen takımında elde edilmesi için kullanılmaktadır. Denklem 4'te verilen dönüşüm matrisiyse, cisim eksenindeki açısal hızlardan Euler açılarının elde edilmesi için kullanılmaktadır.

Dönüşüm matrisleri incelendiğinde sinüs, kosinüs, tanjant ve sekant fonksiyonlarından oluştuğu görülmüştür. Literatürde bu fonksiyonların gerçeklenmesinde CORDIC algoritması [Li, Fang, Li ve Zhao, 2016], Taylor serisi açılımı [Nandi, Prasad, Ananda ve Rekha, 2016] ve ara değer uydurmalı arama tablosu [Mehrgardt, 1983] yöntemlerinin kullanıldığı görülmektedir. CORDIC algoritması 3 adet kayma kaydedicisi, 3 adet toplayıcı-çıkarıcı ve özel bağlantılar içermektedir [Volder, 1959]. Çarpıcısı olmayan donanımlar için geliştirilmiş olan CORDIC algoritması yüksek verimlilikle çalışmaktadır. Tablo yöntemi ise ara değer uydurma prensibiyle çalıştığı için hızlı fakat yüksek hatalı sonuç üretmektedir. Tablo yöntemi, ayrıca, sistemin RAM ihtiyacını artırmaktadır. Bu çalışmada, FPGA'larda dahili çarpıcı donanımlar bulunduğu için ve basit müdahalelerle açılım derecesinin değiştirilebilmesi sayesinde, Taylor serisi açılımı trigonometrik fonksiyon hesaplama yöntemi olarak kullanılmıştır.

YÖNTEM

Taylor Serisi Açılımı

Trigonometrik fonksiyonların gerçeklenmesine sinüs fonksiyonun Taylor açılımıyla başlandı. Taylor açılımının derecesine karar verilirken, varsayılan olarak Taylor açılımını kullanan Simulink trigonometric function bloğu-sin fonksiyonu referans kabul edildi. Sinüs için 7. derece (4 terimli) Taylor açılımının [0,65] derece aralığında kabul edilebilir mertebede hatayla hesaplandığı gözlemlendi. Buradan elde edilen bilgi doğrultusunda kosinüs için de 6. derece (4 terimli) Taylor açılımı uygulandı ve yaklaşık olarak sinüsle aynı açı aralığında girişler için kabul edilebilir hatayla sonuç üretilebildiği gözlemlendi. Bu sebeple [0,90] aralığındaki girişler için sinüs değeri hesaplayan bloğun; [0,45] derece aralığında sinüs Taylor açılımını, [45,90] derece aralığında ise kosinüs Taylor açılımını kullanacağı bir tasarım kararı verildi.

Dijital Sayı Gösterimi

Referans olarak seçilen sinüs hesaplayıcının sonuçlarına mümkün olan en az hatayla ulaşabilmek için Taylor açılımının derecesinin yanında verilerin sayısal gösterim formatının da uygun belirlenmesi gerekmektedir. Sinüs fonksiyonu [-1,1] aralığında değer aldığı için tasarlanacak sinüs hesaplayıcının

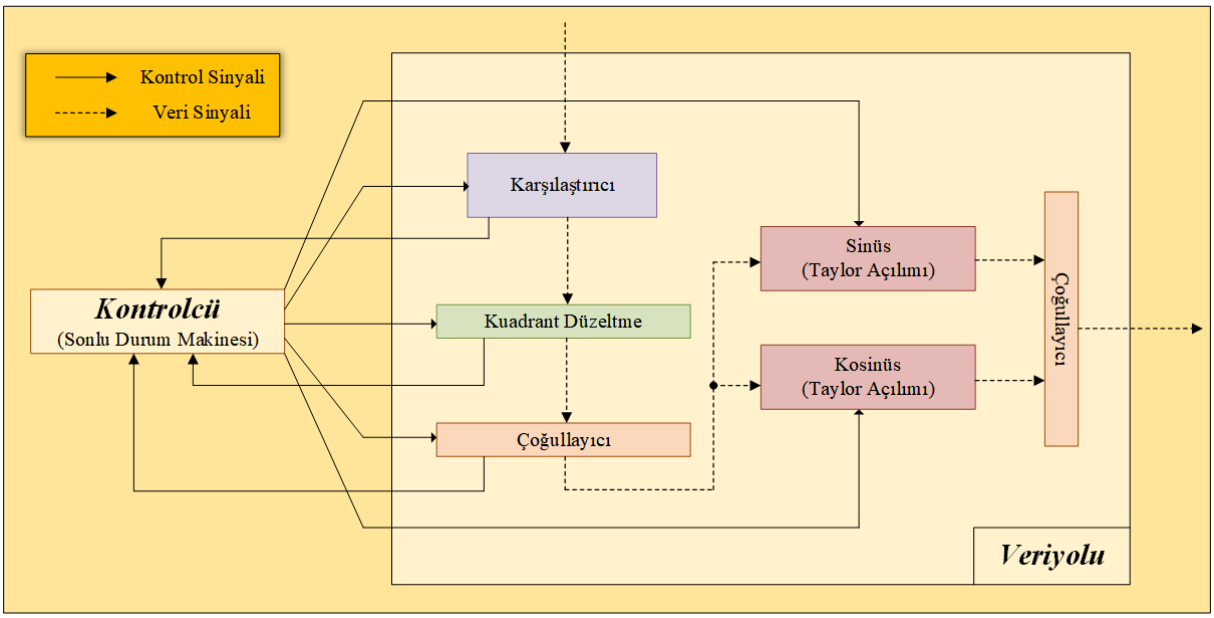
dinamik aralığı sabit ve belirlidir. Bu bilginin varlığında FPGA kaynaklarını daha az tüketeceği de öngörüldüğünden tasarımda sayısal sayı gösterim biçimi olarak sabit noktalı sayılar tercih edilmiştir.

Model Tabanlı Tasarım

Donanım tabanlı gerçekleme için platform olarak seçilen FPGA'lar donanım tanılama dilleri ile programlanmaktadır. Bu noktada, trigonometrik fonksiyonların HDL kodunu elle yazarak tasarlamak yerine Simulink'te MBD yaklaşımıyla beraber gelen modülerlik ve otomatik kod üretimi yöntemlerinin kullanılması tercih edildi. Modülerlik sistem parçalarının ayrı ayrı tasarlanabilmesi, parçalarının tasarım süreçlerinde hata ayıklamayı kolaylaştırması gibi avantajlar sağlarken otomatik kod üretimi de sürecin standartlaştırılması, otomatikleştirilmesi ve güvenilirliğinin arttırılması noktasında önemli katkıda bulunmaktadır.

Standart İşlemci Mimarisi

Simulink ortamında model tabanlı tasarım yaklaşımı kullanılarak yapılan sinüs hesaplayıcı, Şekil 3'te verilen standart işlemci mimarisinde (veriyolu&kontrolcü) [Vahid, 2010] tasarlandı. Sistemin kontrolcü parçası, sinüs ve kosinüs hesaplayıcı bloğun [0,360] derece arasındaki girişler için sonuç üretebilmesi de hedeflendiği için girişe uygulanan açıyı 1. kuadrant'a indirgeyip sonrasında sinüs ya da kosinüs Taylor açılımının uygulanmasını sağlamaktadır. Tasarımın veriyolu parçası ise çarpma, toplama gibi aritmetik işlemleri ve karşılaştırıcı, ve, veya kapıları gibi boolean işlemleri ile hafıza elemanı olarak kullanılan kaydedicilerden oluşmaktadır.



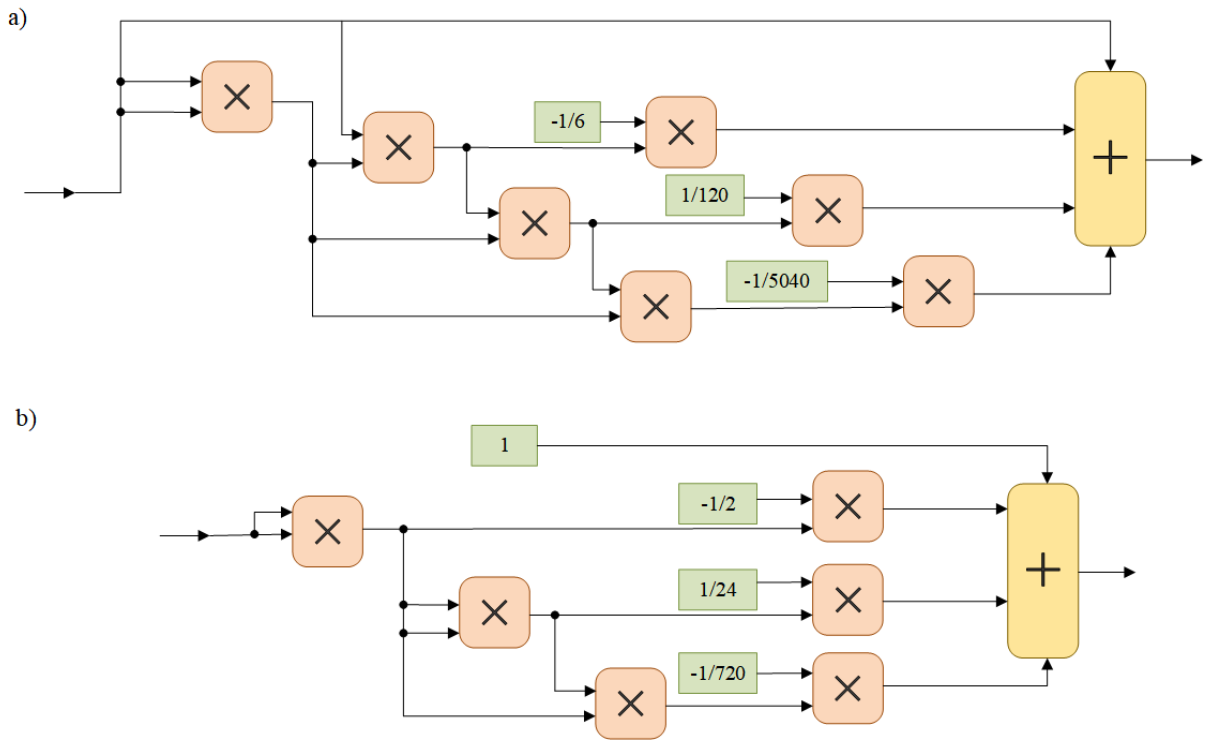
Şekil 3: Standart işlemci mimarisiyle sinüs-kosinüs hesaplayıcı tasarımı

Kaynak Paylaşımı

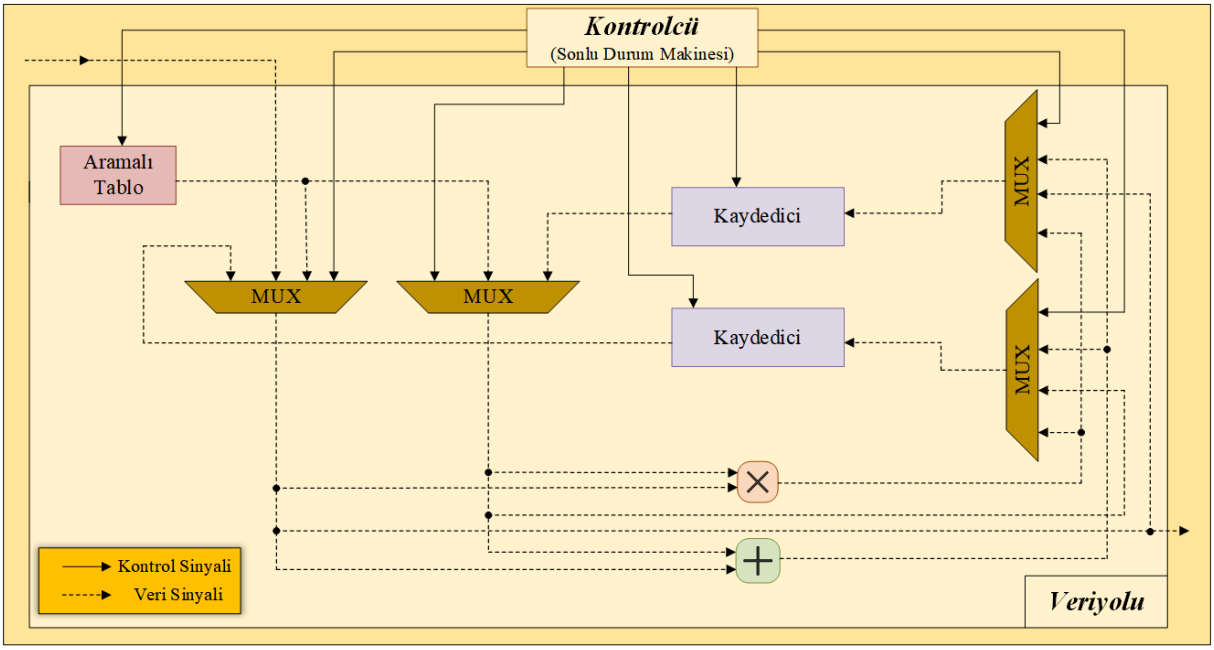
FPGA'ların sahip olduğu kısıtlı kaynakları daha verimli kullanabilmek amacıyla hedeflenen sistemin tasarımında kaynak paylaşımı uygulanmıştır. Zaman-kaynak ödünleşmesinde kaynak kullanımı azaltılarak yapılacak işlemler zamana yayılmıştır. 4 farklı çarpma işleminin paralel olarak gerçekleştirilmesi yerine bu 4 işlemin zamanda sıraya sokularak 1 adet çarpıcı blokla gerçekleştirilmesi tasarımda kullanılan bu yaklaşıma örnek olarak verilebilir. FPGA kaynak kullanımının azaltılmasında, kaynak paylaşımı yaklaşımının yanında, DSP (Digital Signal Processor) birimleri de kullanılmıştır. Bu birimler sinyal işlemede yoğun olarak kullanılan bazı matematiksel operatörleri gerçekleştiren özel donanımlar olup FPGA'ların içerisinde bulunmaktadırlar.

SİSTEM TASARIMI

Sinüs Hesaplayıcı



Şekil 4: Sinüs ve kosinüs fonksiyonları Taylor serisi açılımlarının Simulink tasarımı



Şekil 5: Standart işlemci mimarisinde çarpıcı ve toplayıcı için kaynak paylaşımı uygulaması

Sinüs hesaplayıcı sistem, 1. versiyonunda, veriyolu elemanları kaynak paylaşımı yapılmadan ve fixed-point kelime/kesir biti uzunlukları optimize edilmeden tasarlanmıştır. Şekil 1'de verilen Sinüs, Kosinüs bloklarının Simulink tasarımı Şekil 4'te gösterilmiştir. Sistemin ikinci versiyonunda, veriyolu elemanlarının yapısı aynı tutuldu ve sabit noktalı sayı formatının kelime/kesir biti uzunlukları optimize edilerek tasarım yapıldı. Sistemin üçüncü versiyonunda, sabit noktalı sayı formatının

kelime/kesir biti uzunlukları optimizeyken veriyolu elemanlarında kaynak paylaşımı uygulandı. Bu tasarımda veriyolu elemanlarından çarpma ve toplama/çıkarma blokları, tek çarpma ve tek toplama bloğuna indirgenmiştir. Bu haliyle sistemin veriyolundaki Sinüs, Kosinüs bloklarının Simulink tasarımı sırasıyla Şekil 5'te verilmiştir. Ayrıca dördüncü versiyonda, önceki tasarımda kullanılan çarpma ve toplama blokları DSP'ler ile gerçeklenmiştir. Sinüs hesaplayıcı sistemin 4 versiyonunda da yapılan tasarımlar Simulink'te doğrulandıktan sonra otomatik kod üretimi işlemi uygulanıp Verilog kodları elde edilmiştir.

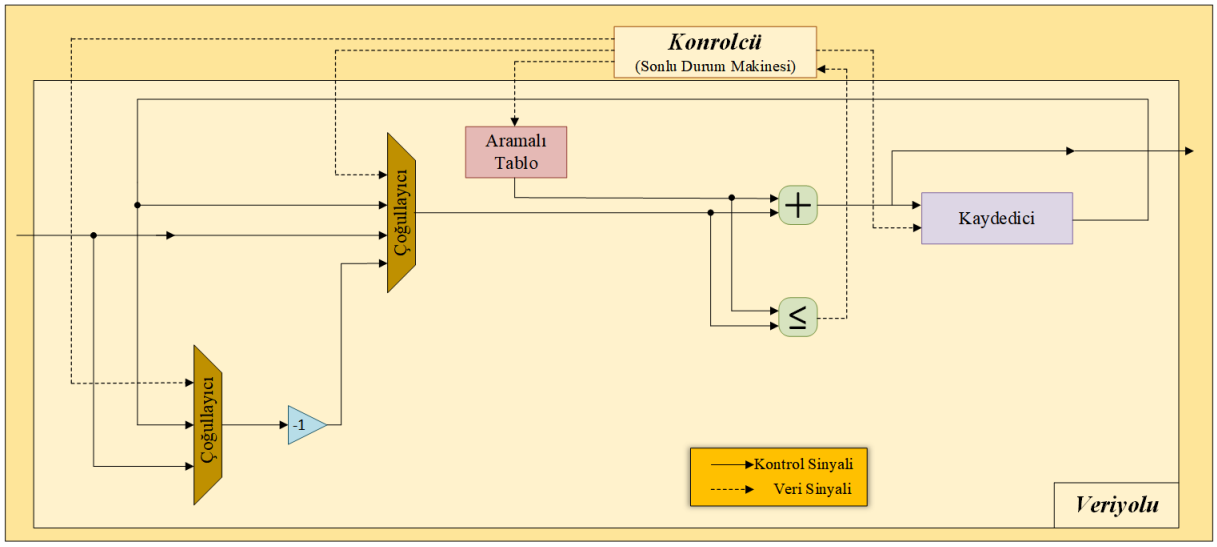
Table 1: 4 farklı sinüs hesaplayıcı tasarımına ait kaynak kullanımları ve zamanlama değerleri

	Versiyon-1	**Versiyon-2**	**Versiyon-3**	**Versiyon-4**
LUT Kullanımı (%)	39.47	24.31	14.41	2.90
Kaydedici Kullanımı (%)	0.59	3.46	0.94	0.91
Hesaplama Süresi (ns)	123.825	27.978	83.202	58.452

Tablo 1'de sinüs hesaplayıcı tasarımının 4 ayrı versiyonu için FPGA kaynak kullanımları (LUT ve Register), hesaplama süreleri ve boştaki zaman payları verilmiştir. Versiyon-1 ile versiyon-2 kaynak kullanımları incelendiğinde, sabit noktalı sayı formatı kelime uzunluklarının optimize edilmesiyle kaynak kullanımı yaklaşık %60 azaltılmıştır. Versiyon-2 ile versiyon-3 için Tablo 1 incelendiğinde açıkça görülmektedir ki paralel çarpma işlemleri yaptırmak yerine tek çarpıcı kullanarak işlemlerin zamana yayılması kaynak kullanımını yaklaşık %68 düşürürken hesaplama süresini %48 artırmıştır. Tablo 1'deki Versiyon-3 ile versiyon-4 için elde edilen değerler kıyaslandığında ise, Taylor açılımı bloklarındaki çarpma işlemlerinin DSP kullanılarak gerçekleştirilmesinin kaynak kullanımını yaklaşık 5 kat azalttığı gösterilmiştir. Öte yandan hazır DSP bölmelerinin kullanılmasıyla hesaplama süresinde yaklaşık %42'lik düşüş olarak gözlenmiştir.

Sinüs-Kosinüs Hesaplayıcı

Kosinüs hesabına geçilmeden önce girilen açının kuadrantını belirleyip gereken düzeltme işlemlerini yapan, Şekil 3'te Karşılaştırıcı ve Kuadrant Düzeltme isimleriyle anılan bloklar tek toplayıcı ile tek bir karşılaştırıcı eleman kullanılarak tekrar tasarlanmıştır. Yapılan bu tasarım Şekil 6'da verilmiştir.



Şekil 6: Girilen açının kuadrantını belirleyip 1. kuadranta indirgeyen yapı

Son durumda girilen açının sinüsü hesaplayan yapı, en üstte bir adet sonlu durum makinesi olarak tasarlanmış kontrolcü ve veriyolundan oluşmaktadır. Veriyolu da kendi içerisinde 2 alt sistem içermektedir. 1. sistem kuadrant belirleyip indirgeme işlemlerini yapan veriyolu elemanları ve

kontrolcüden, 2. sistem ise uygun Taylor açılımlarını gerçekleştien veriyolu elemanları ve kontrolcüden oluşmaktadır. İstenen açının sinüsünü hesaplayan bu sistemin kosinüs hesabı gerçekleştirebilmesi için bahsi geçen 3 adet kontrolcü bloğuna eklemeler yapılması gerekmektedir. Moore makinesi olarak tasarlanan bu kontrolcülere kosinüs hesabına uygun mantık yapılarının eklenmesiyle tüm sistem sinüs ve kosinüs hesabı yapabilir hale gelmektedir.

SONUÇ

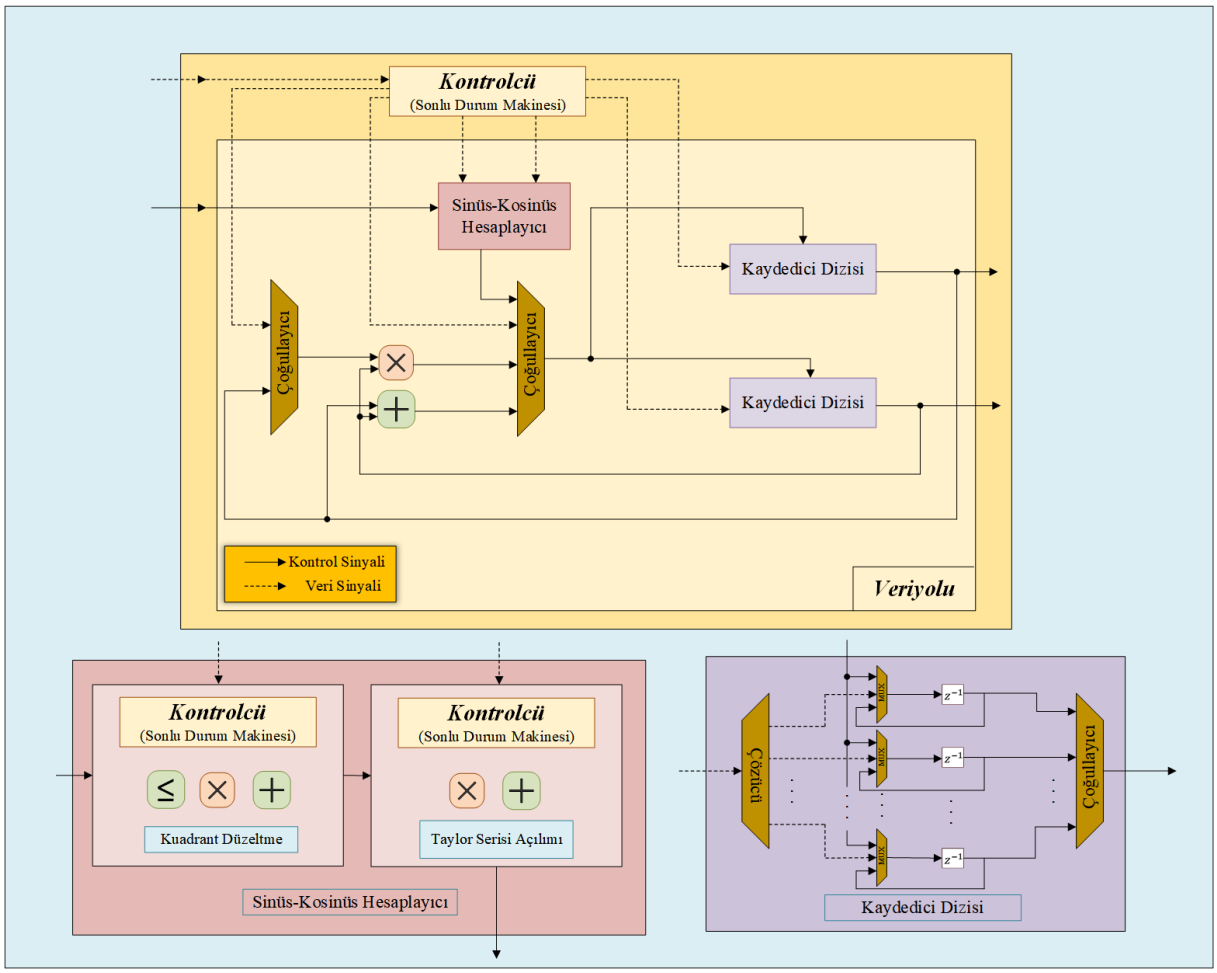
Bu çalışmada, bir sinüs hesaplayıcı model tabanlı tasarım yaklaşımı kullanılarak kontrolcü ve veriyolu yapısında hedef FPGA üzerinde çalışmak üzere Verilog dilinde tasarlandı. Çalışmada sayısal sayı gösterim biçimi olarak seçilen sabit noktalı gösterim kelime ve kesir bit uzunluklarında hedef FPGA olarak seçilen XC7Z010-1CLG400C DSP bölmelerinin özellikleri ve açı girişlerinin dinamik aralığının sınırlı olması optimizasyon kısıtı olarak belirlendi. Tasarımın ilk versiyonunda kaynak paylaşımı yapılmadan paralel çarpma işlemleri yapan bir sinüs hesaplayıcı sistem ortaya konuldu. Kaynak paylaşımı yapılmadığında, FPGA kaynak kullanımının bir sinüs hesaplayıcı için çok yüksek seviyelerde olduğu gösterildi. Tasarımın ikinci versiyonunda ise sabit noktalı gösterim kelime ve kesir biti uzunluklarının optimize edilmesiyle FPGA kaynak kullanımının azaltılması sağlandı. Versiyon-2'deki sistemin veriyolu parçasında bulunan sinüs ve kosinüs taylor bloklarına Şekil 5'teki gibi kaynak paylaşımı uygulanmasıyla versiyon-3'teki tasarım elde edildi. Paralel çarpma, toplama işlemi yapmak yerine bu paralel işlemler, sonlu durum makinesi olarak tasarlanan kontrolcü sayesinde zamanda sıraya sokulup FPGA kaynaklarının daha az kullanılmasını sağlandı. Son olarak, versiyon-4'te, FPGA'nın DSP bölmelerinin içerisinde hazır olarak bulunan çarpıcı blokların kaynak kullanımı ve hesaplama zamanına olan etkisinin gösterilmesi amacıyla Şekil 5'teki çarpıcının DSP bölmeleriyle gerçeklenmesi sağlandı.

Elde edilen bu sonuçların ardından kaynak kullanımının azaltılması amacıyla, Taylor açılımı bloğunda kullanılan kaynak paylaşımı yöntemi kuadrant düzeltme bloğu için de uygulandı. 4 farklı karşılaştırıcı ve 4 farklı toplayıcı kullanmak yerine 1 toplayıcı ile 1 karşılaştırıcıyla tüm kuadrant belirleme ve indirgeme işlemleri gerçekleştirildi. Ardından sistemin halihazırda sahip olduğu veriyolu elemanlarını kullanarak kosinüs hesabı da yapabilmesi için Moore makinesi olarak tasarlanmış kontrolcü bloklara gerekli eklemeler yapıldı. Sonuçta girilen açının sinüsünü ve kosinüsünü hesaplayan bilen tasarımın hedef FPGA üzerindeki kaynak tüketimleri Tablo 2'de verilmiştir. Sistem 20 MHz saat frekansında çalışmakta ve 80 DSP bölmesinin 4 tanesini kullanmaktadır.

Table 2: Sinüs-Kosinüs hesaplayan sistemin hedef FPGA üzerindeki kaynak kullanımları

LUT Kullanımı (%)	3.09
Kaydedici Kullanımı (%)	1.37
Hesaplama süresi (ns)	241.074

Denklem 2'de verilen dönüşüm matrisinin elde edilmesi için Şekil 7'de verilen yapı kullanılmaktadır. Şekil 6'da verilen kuadrant düzelten blokla Şekil 5'te verilen Taylor açılımını gerçekleştiren bloğun kaskat bağlanması ve bu yapıyı yöneten kontrolcüyle sinüs ve kosinüs hesaplayabilen sisteme ek olarak, 1 çarpıcı ve 1 toplayıcı ile işlemlerin sonucunun yazılacağı kaydedici dizisi yapılarının sisteme eklenmesiyle Şekil 7'deki yapı oluşturulmaktadır. Dönüşüm matrisini elde eden bu yapının %4.5 LUT, %1.65 kaydedici, 5 DSP bölmesi kullanımına ve 7 μs civarında hesaplama süresine sahip olması öngörülmektedir.



Şekil 7: Dönüşüm matrisini elde eden yapı. Sinüs-Kosinüs Hesaplayıcı ile Kaydedici Dizisi bloklarının iç yapıları şeklin alt kısmında verilmiştir.

Kaynaklar

Bélanger, J., Venne, P. ve Paquin, 2010. *The What, Where and Why of Real-Time Simulations*, Planet Rt, vol. 1, s. 25–29.

Bouhali, M., Shamani, F., Dahmane, Z.E., Belaidi ve A., Nurmi, J., 2017. *FPGA Applications in Unmanned Aerial Vehicles-A Review.*, In Proceedings of the 13th International Symposium on Applied Reconfigurable Computing (ARC), Delft, The Netherlands.

de Farias A. B. C., Murilo A., ve Lopes R. V., 2021. *Embedded linear model predictive control in field-programmable gate array using register-transfer level implementation and floating-point representation applied to a quadrotor system model*, Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 236.2, s.370-381

Dai, X., Ke, C., Quan, Q. ve Cai, K.-Y., 2021. *RFlySim: Automatic test platform for UAV autopilot systems with FPGA-based hardware-in-the-loop simulations.*, Aerospace Science and Technology

Konomura, R., ve Koichi H., 2012. *Phenox: Zynq 7000 based quadcopter robot.*, 2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14), IEEE

Li, J., Fang, J., Li, B., ve Zhao, Y., 2016. *Study of CORDIC algorithm based on FPGA.*, 2016 Chinese Control and Decision Conference (CCDC), IEEE, s. 4338–4343

Mehrgardt, S., 1983. *Noise spectra of digital sine-generators using the table-lookup method.*, IEEE transactions on acoustics, speech, and signal processing, Vol. 31, s. 1037–1039

Nandi, S., Prasad, S., Ananda, C., ve Rekha, S., 2016. *Fixed point implementation of trigonometric function using Taylor's series and error characterization.*, 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, s. 442–446

Vahid, F., 2010. *Digital design with RTL design, VHDL, and Verilog.*, John Wiley & Sons

Volder, J. E., 1959. *The CORDIC trigonometric computing technique.*, IRE Transactions on electronic computers, s. 330–334

Zipfel, P.H. 2007.. *Modeling and Simulation of Aerospace Vehicle Dynamics.*, Virginia: American Institute of Aeronautics and Astronautics