T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ
ANABİLİM DALI

FPGA İLE GERÇEK-ZAMANLI RUNGE-KUTTA MODEL ÖNGÖRÜLÜ KONTROL

DOKTORA TEZİ

BEDRİ BAHTİYAR

DENİZLİ, MAYIS - 2015

T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ
ANABİLİM DALI

FPGA İLE GERÇEK-ZAMANLI RUNGE-KUTTA MODEL ÖNGÖRÜLÜ KONTROL

DOKTORA TEZİ

BEDRİ BAHTİYAR

DENİZLİ, MAYIS - 2015

KABUL VE ONAY SAYFASI

BEDRİ BAHTİYAR tarafından hazırlanan "**FPGA İLE GERÇEK ZAMANLI RUNGE-KUTTA MODEL ÖNGÖRÜLÜ KONTROL**" adlı tez çalışmasının savunma sınavı 11.05.2015 tarihinde yapılmış olup aşağıda verilen jüri tarafından oy birliği ile PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI DOKTORA TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	İmza
Danışman Prof.Dr. SERDAR İPLİKÇİ	.....
Üye Prof.Dr. M.MELİH İNAL	.....
Üye Doç.Dr. SEZAİ TOKAT	.....
Üye Doç.Dr. KADİR KAVAKLIOĞLU	.....
Üye Yrd.Doç.Dr. SELAMİ BEYHAN	.....

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 20/05/2015. tarih ve ..18/15..... sayılı kararıyla onaylanmıştır..

.....

Prof. Dr. Orhan KARABULUT

Fen Bilimleri Enstitüsü Müdürü

Bu tez çalışması Pamukkale Üniversitesi tarafından 2013FBE006 nolu proje ile desteklenmiştir.

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğini beyan ederim.

BEDRİ BAHTİYAR

ÖZET

FPGA İLE GERÇEK-ZAMANLI
RUNGE-KUTTA MODEL ÖNGÖRÜLÜ KONTROL
DOKTORA TEZİ
BEDRİ BAHTİYAR
PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

(TEZ DANIŞMANI: PROF. DR. SERDAR İPLİKÇİ)

DENİZLİ, MAYIS - 2015

Bu tez çalışmasında, Altera firmasının DE2-115 FPGA eğitim ve geliştirme seti üzerinde bulunan Cyclone IV tip FPGA üzerinde RKMPC yönteminin (İplikçi 2013) gerçeklenmesi ile elde edilen kontrolör kullanılarak, kararsız, çok hızlı ve Doğrusal Olmayan bir sistem olan EMLS'nin kontrolü sağlanmıştır. Bununla birlikte, geleneksel Doğrusal Olmayan MPC olarak kabul edilen ve (Plucenio ve diğ. 2007)'de önerilen yöntemi uygulayan algoritma FPGA üzerinde gerçeklenerek, RKMPC gerçeklemesi ile ilgili çalışmalar tekrarlanmıştır. Her iki yönteme ait FPGA gerçeklemeleri ile oluşturulan iki farklı kontrolör birbirleri ile karşılaştırılmış ve NMPC yöntemi ile kabul edilebilir bir kontrol performansı elde edilebilmesine rağmen, RKMPC yöntemi ile hem çıkış işaretinin referans işaretini izlemesi, hem kontrol işaretinin yeterliliği, hem de durum değişkenlerinin değişimleri açısından çok daha iyi bir kontrol performansı sergilendiği gösterilmiştir. Ayrıca RK tabanlı parametre kestirimi algoritması (İplikçi 2013), RKMPC algoritması ile bütünleşik olarak FPGA üzerinde gerçeklenmiş ve $\theta = {}^{k}/_{m}$ şeklinde seçilen parametrenin değerinin kestirimi sırasında Model-Uyarlamalı Öngörülü Kontrol uygulaması başarılı bir şekilde gerçekleştirilmiştir.

ANAHTAR KELİMELER: Doğrusal Olmayan Model Tabanlı Kontrol, FPGA, Optimal Kontrol, Model Öngörülü Kontrol, Gömülü Sistemler

ABSTRACT

REAL TIME RUNGE-KUTTA MODEL PREDICTIVE CONTROL WITH FPGA

PH.D THESIS

BEDRİ BAHTİYAR

PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE

ELECTRICAL AND ELECTRONICS ENGINEERING

(SUPERVISOR: PROF. DR. SERDAR IPLIKCI)

DENİZLİ, MAY 2015

In this thesis, an EMLS, which is unstable, very fast and nonlinear system, has been controlled with using a controller obtained by implementation of RKMPC (Iplikci 2013) on a Cyclone IV type FPGA, which has been existed in Altera DE2-115 Education and Development Board. Besides, the method, introduced in (Plucenio vs 2007), has been assumed as the traditional NMPC and then the studies on RKMPC implementation has been repeated for FPGA implementation of the NMPC. After comparison of these two controllers, it has been shown that, although the NMPC implementation has exhibited an acceptable control performance, the RKMPC implementation has shown better control performance than the NMPC implementation, in terms of tracking the reference signal, efficiency of the control signal and the variations of the state variables. Afterwards, the RK based parameter estimation algorithm (Iplikci 2013), integrated in the RKMPC, has been embedded to the FPGA and an adaptive control of EMLS has been successfully realized while estimation of a parameter which has been chosen which has been formulated as $\theta = {}^{k}/_{m}$.

KEYWORDS: Nonlinear Model Based Control, Field-Programmable Gate Arrays, Optimal Control, Model Predictive Control, Embedded Systems

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

SEMBOL LİSTESİ

$\mathbb{R}$: Reel Sayılar Kümesi

$\boldsymbol{\lambda}$: Giriş İşaretindeki Değişimleri Cezalandırma Parametresi

Δ : Değişim

$\boldsymbol{\eta}$: Hessian Matrisini Pozitif Tanımlı Yapacak Skaler Büyüklük

$\boldsymbol{\delta}$: Değişim

$\boldsymbol{\partial}$: Kısmi Türev

$\boldsymbol{\mu}$: $0 < \mu \leq 1$ Şeklinde Tanımlanan Skaler Büyüklük

$\boldsymbol{\theta}$: Kestirimi Yapılacak Parametre/ler

$\boldsymbol{\Omega}$: Direnç Ölçü Birimi - Ohm

$\boldsymbol{\beta}$: EMLS'de Küre Mıknatısın Bobine Olan Uzaklığına Bağlı Parametre

$\boldsymbol{\gamma}$: EMLS'de Akıma Bağlı Parametre

$\boldsymbol{\alpha}$: EMLS'de Çıkış İşareti Taban Seviyesi

ÖNSÖZ

Uzun ve zorlu bir süreç olan doktora eğitimim boyunca bana desteklerini esirgemeyen başta danışman hocam Sayın Prof.Dr. Serdar İPLİKÇİ olmak üzere Tez İzleme Komite Üyelerim, Sayın Prof.Dr. M.Melih İNAL, Doç.Dr. Sezai TOKAT, Doç.Dr. Kadir KAVAKLIOĞLU ve Yrd.Doç.Dr. Selami BEYHAN'a, ayrıca her zaman yanımda olan, sevgi ve desteğini üzerimde hissettiğim sevgili eşim Sevinç, çocuklarım Bilge ve Umut'a, bu günlere gelmemde büyük pay sahibi olan annem Behiye ve babam Yusuf'a tüm kalbimle sevgi, saygı ve minnetlerimi sunarım.

1. GİRİŞ

Bu tez çalışmasında, Runge-Kutta Model Öngörülü Kontrol (Runge-Kutta Model Predictive Control - RKMPC) yöntemi, Alanda Programlanabilir Kapı Dizisi (Field Programmable Gate Array - FPGA) üzerinde gerçeklendikten sonra gerçek-zamanlı bir Doğrusal Olmayan sistemin kontrolünün sağlanması hedeflenmiştir. Bu sayede RKMPC algoritmasının gömülü olarak FPGA üzerinde gerçeklenebilirliği ve böyle bir kontrolörün oldukça hızlı, kararsız ve Doğrusal Olmayan bir sistem olan Elektromanyetik Askı Sistemi (Electromagnetic Levitation System- EMLS) üzerindeki kontrol performansı incelenmiştir. Elde edilen sonuçlar geleneksel Doğrusal Olmayan Model Öngörülü Kontrol (Nonlinear Model Predictive Control – NMPC) (Plucenio ve diğ. 2007) ile karşılaştırılmış ve bunlara ek olarak kontrol eylemi sırasında çevrimiçi, (5.5) ile verilen sistem durum denklemleri içerisindeki $\theta = k/m$ parametresinin kestirimi gerçekleştirilmiştir.

1.1. Model Öngörülü Kontrolün Tarihsel Gelişimi

Model Öngörülü Kontrol (Model Predictive Control –MPC) düşüncesinin ortaya çıkışı 1960'lı yıllarda başlamış olsa da (Garcia 1989), bu alana ilgi 1978 yılında yapılan bir çalışmanın yayınlanması ile başlamıştır (Richalet ve diğ. 1978). Bu çalışmada, kimya endüstrisinde, çok değişkenli bir sistemin, referans işaretini izlemesini sağlamak üzere MPC yöntemi kullanılmıştır. Bunu izleyen süreçte Dinamik Matris Kontrol (Dynamic Matrix Control - DMC) (Cutler, 1980) ve Genelleştirilmiş Öngörülü Kontrol (Generalized Predictive Control - GPC) yöntemlerinin ilk geniş kapsamlı sunumu gerçekleştirilmiştir (Clarke 1987[a,b]). Bu çalışmalar; MPC yönteminin ilerleyen ufuk özelliği ile, sistemin durum ve çıkışları üzerindeki kısıtları dikkate alabilmesi ve zaman düzleminde formüle edilebilmesi gibi özellikleri nedeniyle diğer kontrol yöntemlerine üstünlük sağladığını göstermiştir. Daha sonraları literatürde pek çok MPC benzeri yöntem önerilmiştir (De Keyser 1985; Soeterboek 1992). Bu yöntemler; minimum-fazlı

olmayan sistemler, açık-çevrimi kararsız sistemler ve değişken ölü-zamanlı ve/veya parametreli sistemler gibi pek çok endüstriyel sistemin kontrolünde başarılı bir şekilde kullanılmışlardır (Clarke 1989; Camacho 1993). Önerilen yöntemler özellikle GPC yöntemine benzemektedir ve hala tüm MPC yöntemleri aynı fikre dayanmaktadır: Her örnekleme adımında, önce kontrol edilecek sisteminin matematiksel modeli kullanılarak, belirli bir kestirim ufku boyunca, bir aday kontrol işaretine sistemin vereceği cevaplar hesaplanır. Daha sonra bu kestirim çıkışları ile aynı kestirim ufkundaki referans işareti değerleri arasındaki farkların karesinin optimize edilmesi ile sisteme uygulanacak yeni kontrol işareti hesaplanır. Elde edilen kontrol işaretinin sisteme uygulanması ile bir sonraki örnekleme adımına kadar işlem sona erer. Dolayısıyla MPC açısından bakıldığında, kontrol edilecek sistemin matematiksel modeli büyük önem taşımaktadır (İplikçi 2013).

İlk MPC yöntemleri, sistem modelinin darbe cevabı (Richalet ve diğ. 1978) ve basamak cevabına (Cutler ve Ramaker 1980) dayanmaktadır. Bu yöntemler kararlı sistemler için yeterli performans gösterebiliyor olsalar bile kararsız sistemler için kullanılamazlar. Bu sorunun üstesinden gelmek amacıyla Control Auto-Regressive and Moving-Average (CARMA) (De Keyser ve Cauwenberghe 1985) ve Control Auto-Regressive and Integrated Moving-Average (CARIMA) (Clarke ve diğ. 1987[a,b]) modelleri önerilmiştir. Bu modellere yönelik benzetimler sonucunda önerilen modellerin gürbüz ve o ana kadar kabul edilen uyarlamalı kontrolörlere göre oldukça üstün olduğu, kestirim ufkunun küçük seçilmesi durumunda bir mikroişlemciye yüklenebileceği belirtilmiştir (Clarke ve diğ. 1987[a,b]). Bu önerilen MPC yöntemleri ayrık-zamanlı olsalar bile, sürekli-zamanlı tipleri de literatürde bulunmaktadır (Demircioglu ve Gawthrop 1991; Gawthrop ve Demircioglu 1989). Tüm bu GPC yöntemleri doğrusal sistemler için geliştirilmiştir ve dolayısıyla problem, analitik olarak çözümü yapılabilecek olan optimizasyon problemine dönüşmüştür. Oysaki endüstride ve pek çok bilimsel çalışmada kontrol edilmesi gereken sistemler çoğunlukla Doğrusal Olmayan dinamiklere sahiptir. Doğrusal Olmayan sistemlerin kontrolünde GPC yöntemlerini kullanabilmek için uygulanabilecek yaklaşımlardan birisi, bir çalışma noktası etrafında sistemin dinamiklerinin doğrusallaştırılması ve ardından GPC yönteminin uygulanmasıdır. Ancak, bir

çalışma noktası civarında doğrusal olmama derecesi yüksek sistemlerde, çalışma bölgesinin tümüne dağılmış Doğrusal Olmayan işlemlerde ve referans işaretinin tüm çalışma bölgesinde sıklıkla değiştiği durumlarda, doğrusallaştırma yaklaşımı etkinliğini kaybetmektedir. Bu nedenle Doğrusal Olmayan sistem modelini kullanan, Doğrusal Olmayan model öngörülü kontrol (Nonlinear Model Predictive Control - NMPC) yöntemi önerilmiştir (Camacho ve Bordons 2007[a,b]; Henson 1998). Önerilen NMPC tekniği, her örnekleme adımında kısıtlı Doğrusal Olmayan bir optimizasyon probleminin çözümünü gerektirir. Ancak bu tür probleme ait global bir çözümün bulunması genellikle çok zor ve uzun zaman gerektirmektedir. Bunun yerine uygun alt çözümlerin kullanılabileceği ve hatta bir yerel minimum bulunmasının bile gerekli olmadığı, her örnekleme adımında yapılması gereken asıl işlemin amaç fonksiyonda yeterli azalmayı sağlayacak bir çözümün bulunması olduğu, ortaya koyulmuştur (Scokaert ve diğ. 1999; Maciejowski 2002).

NMPC yöntemi kontrol edilecek Doğrusal Olmayan sistemin modeline bağlıdır. Fakat yeterli bir Doğrusal Olmayan model geliştirmek oldukça zor olabileceği gibi böyle bir sistemi tam olarak ifade edebilecek bir model şekli de yoktur. Genel olarak NMPC döngülerinde kullanılan üç tip Doğrusal Olmayan sistem modeli vardır: Temel Model (Fundamental Model), Deneysel Model (Empirical Model), Hibrit (Hybrid Model). Temel modeller, temel prensipler de denilen; sisteme kütle, enerji ve moment dengesi uygulanması sonucunda elde edilen Adi Diferansiyel Denklemleri (Ordinary Differential Equations – ODEs) ve bazı durumlarda ek cebirsel eşitliklerden oluşur. Deneysel Modeller ise sistemin giriş/çıkış verileri üzerinden elde edilmektedir. Fakat Doğrusal Olmayan sistemlerde süperpozisyon prensibi olmadığından giriş/çıkış verileri üzerinden bir model oluşturmak çok zordur. Doğrusal bir sistemde, sisteme uygulanan herhangi bir basamak işaretine sistemin verdiği cevaba bakarak, sistemin genel basamak cevabı öğrenilebilir. Fakat Doğrusal Olmayan bir sistemde, sistemin basamak cevabını öğrenmek için değişik aralıklarda çok sayıda basamak işareti uygulayıp bu işaretlere sistemin verdiği cevapları topluca değerlendirmek gerekir (Camacho and Bordons 2003). Literatürde NMPC döngüsünde kullanılan pek çok deneysel model bulunmaktadır: Polinom ARMA modelleri (Hernandez ve Arkun 1993), Hammerstein modelleri (Fruzzetti ve diğ. 1997), Volterra modelleri (Doyle ve diğ.

1995; Genceli ve Nikolaou 1995; Gruber ve diğ. 2010; Maner ve diğ. 1996), Wiener modelleri (Cervantes ve diğ. 2003; Norquay ve diğ. 1998), yapay sinir ağları (Lawrynczuk 2007; Piche ve diğ. 2000), bulanık mantık modelleri (Cho ve diğ. 1999; Roubos ve diğ. 1999) gibi yazılımsal bilgisayar araçları kullanan modeller ve destek vektör makineleri gibi makine öğrenmesi araçları (İplikçi 2006, 2010; Xi ve diğ. 2007) bunlara örnek olarak verilebilir. Son olarak hibrit modeller ise her iki yaklaşımı da kullanarak elde edilen modellerdir (İplikçi 2010). NMPC yönteminin kullanımını zorlaştıran en önemli öğelerden biri olan Doğrusal Olmayan sistem modelinin çıkarılmasına yönelik olarak hem temel metot hem de deneysel metot kullanan pek çok araç bulunmakta olup kullanımı da gün geçtikçe artmaktadır (Camacho 2007[a]).

NMPC yönteminin MPC yöntemine göre temel üstünlüğü sistemin Doğrusal Olmayan dinamiklerini değerlendirebilmesidir. Temel doğrusal MPC kavramı içinde NMPC kullanımına karşı herhangi bir şey olmadığı için MPC yönteminin Doğrusal Olmayan sistemlere doğru bir şekilde yayılması en azından kapsam olarak kolaydır. Buna rağmen bu yönelim, problemin kolayca halledilebileceği anlamı taşımaz ve bu tür modellerin kullanımı sırasında karşılaşılabilecek pek çok zorluk bulmaktadır. Bunlardan bazıları şu şekilde sıralanabilir:

- Deneysel verilerden Doğrusal Olmayan modellerin elde edilebilmesi konusu halen tamamlanmış bir konu değildir. Doğrusal Olmayan sistemlerin tanımlanmasında kullanılacak teknikler konusunda eksiklikler vardır. Örneğin; sinir ağları veya Volterra dizileri bir genel form halinde problemi çözebilecek gibi görünmemektedir. Diğer taraftan, kütle ve enerji dengelerinden yararlanılan temel prensipler ile modele ulaşmak ise genellikle mümkün değildir.

- Optimizasyon problemi konveks değildir ve çözülebilirliği Karesel Problem (Quadratic Problem - QP) ile karşılaştırıldığında çok daha zordur. Yerel optimuma göre ortaya çıkan problemler sadece kontrol kalitesini etkilemekle kalmaz, aynı zamanda, kararlılık problemlerine de yol açar.

- Optimizasyon probleminin çözümündeki zorluk, hesaplama zamanı üzerinde oluşacak önemli bir artış anlamına gelir. Bu durum NMPC yönteminin daha yavaş sistemlerde kullanılması şeklindeki bir kısıtlamaya neden olur.

Yukarıda madde imleri şeklinde sunulan problemlerden bazıları kısmen çözülmüştür ve NMPC yoğun olarak çalışılan bir araştırma alanı haline dönüşmüştür (Camacho ve Bordons 2007[a]).

Önerilen NMPC tekniklerinin pek çoğunun ayrık-zamanlı sistemler için geliştirilmiş olmasına rağmen, üzerinde çalışılan doğal ya da endüstriyel sistemler çoğunlukla sürekli-zamanlı sistemleridir. Bu nedenle Doğrusal Olmayan optimal kontrol probleminin (Nonlinear Optimal Control Problem – NOCP), ayrıklaştırma (discretization) veya yaklaşıklık (approximation) yöntemleri Doğrusal Olmayan programlama problemine (Nonlinear Programming Problem – NLP) dönüştürülmesi ve ardından bir NLP kullanılarak çözülmesi gerekir.

Bu amaçla, Collocation on Finite Elements (Kawathekar ve Riggs 2007), Multiple Shooting (Schafer ve diğ. 2007) ve bunların birlikte kullanımı (Tamimi ve Li 2010) gibi, Lyapunov'un doğrudan metodu temeline dayanan birkaç yöntem önerilmiştir. Bundan farklı bir yaklaşım olarak, temel prensip ile elde edilen temel modellerdeki ODE'lerin ayrıklaştırılması ile sürekli-zamanlı sistemlerinin ayrık-zamanlı modelleri kullanılmıştır. Doğrudan ve dolaylı Euler metotlarını kullanan bir çalışma (Sistu ve Bequette 1996) buna örnek olarak verilebilir.

Doğrusal Olmayan sistemlerin kontrolü için yakın zaman önce RKMPC yöntemi (İplikçi 2013) önerilmiştir. Bu yöntemde, Runge-Kutta (RK) modeli olarak adlandırılan, Doğrusal Olmayan sürekli-zamanlı sistemin bir ayrıklaştırılmış modeli klasik dördüncü-dereceden RK algoritması ile elde edilmiştir. RK modeli olarak adlandırılan bu model hem ileriye yönelik kestirimlerin hem de türev hesaplarının yapılması amacıyla MPC döngüsü içinde kullanılmıştır. Farklı sistem modelleri üzerinde yapılan benzetim çalışmalarının sonuçları, bu modelin kullanımı ile -ayrı ayrı veya birleşik halde- MPC, durum kestirimi ve/veya çevrimiçi parametre kestirimi yapabilecek bir uyarlamalı Doğrusal Olmayan öngörülü kontrolörün elde edildiği gösterilmiştir (İplikçi 2013).

1.2. MPC Uygulamaları ve Gömülü sistemler

Endüstride otomatik kontrol süreçlerinde, yerinde ayarlanabilen pek çok yöntem kullanılmaktadır. Bu tür kontrolörler ile yapılan ayarlamaların ikinci-dereceden sistem modeline eşdeğer olduğu düşünülebilir fakat bu şekilde elde edilmiş bir model çoğunlukla sistemi tam olarak ifade etmekten uzaktır. Kurallar ve ayarlama amaçlarının "çalışıyorsa iyidir" şeklinde düşünülmesi, bazı yöntemlerin –örneğin PID- neden bu kadar çok tercih edildiğini açıklamaktadır. Geçmişte üretim endüstrisinde kontrol problemleri %80-90 oranında bu tür yöntemler ile çözülebilmekteydi, fakat gün geçtikçe çok değişkenli, karşılıklı etkileşimli, kısıtlamalı, Doğrusal Olmayan ve optimal olma özelliğinin sadece hata değişimi ya da kapalı-çevrim zaman cevabı üzerinde arandığı sistemler karşımıza çıkmaktadır. Bu performans karakteristiklerine ulaşabilmek için PID ve benzeri yöntemler üstün yeterliliklerini kaybetmiş ve MPC gibi başka kontrol yöntemlerini ön plana çıkarmıştır (Richalet 1993).

1970'lerden başlayarak endüstriyel kontrol süreçlerinde artan bir şekilde kullanılan MPC, optimizasyon işlemi kapsamında bulunan kısıtlamalı kontrol problemleri ile ilgilidir ve çeşitli MPC araştırmaları (Morari ve Lee 1999; Qin ve Badgwell 2003; Algöwer ve diğ. 2004; Morari ve Baric 2006), MPC'nin en etkileyici özelliğinin kısıtları değerlendirebilmedeki belirgin yeteneği olduğunu göstermiştir. Bu özellik dinamik sistemin gelecek davranışlarına yönelik olarak model tabanlı öngörülerden kaynaklanmaktadır. Bu öngörüler yapılırken kısıtların gelecekteki giriş, çıkış ve durum değişkenlerine uygulanması ile problem, çevrimiçi QP veya NLP problemi halini alır (Xi 2013).

Kısıtlamalı kontrol problemlerinin çözümüne yönelik olarak etkin bir kontrol tekniği olan MPC dünya genelinde endüstriyel kontrol süreçlerinde başarılı bir şekilde uygulanmış olsa da, aşağıda sıralanan bazı sınırlamalar üzerinde durmak gerekir (Xi 2013):

1- Mevcut endüstriyel MPC algoritmaları, dinamikleri yavaş olan sistemlere ve yüksek performanslı bilgisayarlarla birlikte oluşturulmuş ortamlara uygundur ve bu durum MPC uygulamalarının daha geniş alanlara ve koşullara yayılmasının önündeki en büyük engeldir. Mevcut endüstriyel MPC

algoritmalarında, model ve kısıtlamalarla birleştirilmiş bir optimizasyon probleminin her örnekleme adımında çevrimiçi şekilde çözülmesi gerekir. Fakat bu durum, ağır bir hesaplama yükü ve daha fazla çözümleme zamanı sonucunu doğurur (Xi 2013).

2- Uygulanan sistemler açısından bakıldığında, mevcut MPC algoritmalarının doğrusal veya sözde-doğrusal (quasi-linear) sistemler ile sınırlı olduğu görülür. Doğrusal Olmayan sistemlere yönelik olarak MPC ürünleri geliştirilmiş olsa da, NMPC uygulamalarının adedi, MPC uygulamalarının %2'sidir ve bu oran NMPC yönteminin etkin kullanımdan çok uzak olduğunu gösterir (Qin 2003). Arıtma, petrokimya vb. gibi doğrusal olmama düzeyleri daha az olan üretim endüstrilerinde bile MPC uygulamaları sınırlıdır. Kaldı ki, doğrusal olmama derecesi daha yüksek endüstrilerde MPC uygulamaları nadiren görülür. Bunun başlıca iki nedeni vardır: Birincisi; Doğrusal Olmayan bir sistemin NMPC için kullanılacak olan tam bir modelini elde etmek, genellikle zaman alıcı ve pahalı bir iştir. İkincisi, NMPC içindeki Doğrusal Olmayan kısıtlamalı optimizasyon probleminin nümerik olarak çözümü genellikle uzun zaman alır. Oysa, kontrol sisteminin her örnekleme adımında bozucu etkileri baskılayabilecek ve referans işaretini doğru takip edebilecek bir kontrol işaretini hazırlaması gerekmektedir. Bu nedenlerle Doğrusal Olmayan MPC, her ne kadar akademik araştırmalarda oldukça aktif bir alan olsa da endüstriyel uygulamalarda halen emekleme aşamasındadır (Klatt ve Marquadt 2009).

3- Mevcut endüstriyel MPC algoritmalarının çoğu, basamak veya darbe cevabı modeli gibi endüstride kolayca elde edilebilen parametrik olmayan modeller kullanır ve çevrimiçi kısıtlamalı optimizasyon problemini çözmek suretiyle optimal kontrol gerçekleştirir. Analitik çözümler kullanarak, tasarım parametreleri ile sistem performansı arasındaki ilişkiyi bulmak zordur ve kısıtlamalı sistemler için analitik çözüm yoktur. Bu tür algoritmalardaki tasarım parametrelerinin doğru bir şekilde belirlenmesi ve çok sayıda benzetimler yapılarak test edilmesi gerekir. Bu nedenle, ön çalışmaların yüksek maliyetli oluşunun yanı sıra, alandaki teknisyenlerin deneyimlerinin MPC uygulamalarının başarısında anahtar rol oynaması ve MPC teknolojisinin gerçeklenmesi ve bakımının yüksek seviyeli uzmanlık gerektirmesi, daha fazla uygulamalar yapılmasını engellemektedir (Xi 2013).

Üretim, havacılık, uzay araştırmaları gibi alanlarda çokça karşılaşılan hızlı dinamikleri olan sistemlerin kontrolünde, kontrolörün işlem hızı çok önemli hale gelmektedir. Çünkü her örnekleme adımı içinde kısıtlamalı optimizasyon probleminin çözümü yapılarak uygun bir kontrol işaretinin bulunması gerekmektedir. Bu ise çoğu zaman ağır bir işlem yükünü beraberinde getirmekte ve yüksek hızlı işlemcilere ihtiyaç duymaktadır. Bu nedenle yakın zamana kadar yapılan çalışmaların pek çoğu bilgisayar üzerinde gerçekleştirilen benzetimlerden veya yavaş dinamikleri olan sistemlerin bilgisayarla kontrolünden oluşmaktadır. Fakat endüstride her ortam bilgisayarların kullanımına uygun değildir ve yüksek maliyet, gerçekleme ve hata ayıklama işlemlerindeki karmaşıklık, yüksek dereceli bakım gereksinimleri vb. yönleri mevcut MPC algoritmalarının basit endüstriyel sistemlere uygulanmasına bile engel oluşturmaktadır. Bu problemlerin üstesinden gelebilmek için bir yonga ile kolayca gerçeklenebilen, kısıtlamalı optimizasyon kontrolü için uygun hale getirilmiş kısıtlamalı MPC kontrolörler geliştirilmesi gerekmektedir. Böyle bir kontrolör, kısıtlamalı kontrol problemlerini de değerlendirebiliyor olması nedeniyle, şimdiye kadar maliyet, hata ayıklama ve gerçekleme konuları açısından haklı bir başarıya sahip olan ve endüstride en çok kullanıma sahip PID kontrolörleri gibi, kullanıcılar tarafından evrensel bir kontrolör olarak kabul görebilir (Xi 2013).

Bu amaçla yapılmış ve literatürde göreceli olarak az sayıda olan çalışmalar incelendiğinde, işlemci teknolojisindeki gelişmelere de paralel olarak, MPC algoritmalarındaki işlem yükünün olabildiğince aynı anda farklı işlemleri yapabilme yeteneğine sahip ortamlara dağıtılmaya çalışıldığı görülmektedir. 1999 yılında Sandoz ve diğ. tarafından yapılan çalışmada işlem yükünün farklı işlemciler tarafından paylaşılması anlamına gelen Dağıtılmış Kontrol Sistemleri (Distributed Control System - DCS) kullanılarak doğrusal ve Doğrusal Olmayan sistemler için bir MPC kontrolör ilk örneği (prototype) geliştirilmeye çalışılmıştır (Sandoz ve diğ. 1999). Bu çalışmada Foxboro I/A adlı DCS sisteminde iki blok üzerinde işlemlerin bölüştürülmesi ile elde edilen bir kontrolörün sınırlı da olsa doğrusal sistemler üzerinde uygulanabileceği ortaya konmuştur. Fakat ne yazık ki Doğrusal Olmayan sistemler için böyle bir ilk örneğin ortaya konulmasında mevcut bilgi ve teknolojilerin yeterli olmadığı belirtilmiştir(Sandoz ve diğ. 1999). Daha sonraları paralel çalışan Programlanabilir Denetleyicilerin (Programmable

Logic Controller - PLC) oluşturduğu ağlar, mikro bilgisayar temelli yapılar DCS olarak kullanılmış olmasına rağmen bunların performanslarının yeterli olmadığı ve MPC uygulamalarında bu tip yapıların tek-girişli tek-çıkışlı (Single Input Single Output - SISO) sistemlerle sınırlı olduğu belirtilmiştir. Çok-girişli çok-çıkışlı (Multiple-Input Multiple-Output - MIMO) sistemlerde SISO sistemlere göre daha çok hesaplama gerektiğinden, hesaplama özellikleri çok daha iyi olan gerçek-zamanlı çoklu işlemcilerin kullanımı önerilmiştir. Bu amaçla 25MHz osilatör frekansında çalışan dört adet Motorola 68040 işlemcisinin bulunduğu böyle bir çoklu işlemci yapısı kullanılarak, çeşitli hesaplama algoritmalarının gerçekleştirilmesinde algoritmanın farklı bölümlerini farklı işlemciler üzerinde paralel olarak işletmek suretiyle hesaplama yeteneklerinde elde edilen iyileşmeler ortaya konulmuştur (Hassapis 2003). 2005 yılında Bleris ve diğ. tarafından yapılan bir çalışmada, içerisinde Motorola'nın 32-bit MPC555 işlemcisi bulunan yüksek performanslı bir mini bilgisayar olan phyCORE-MPC555 kartı üzerinde MPC algoritması gerçeklenmiş ve oldukça yavaş sayılabilecek 1 sn. örnekleme adım aralığında processor-in-the-loop co-simulation ile test edilmiştir. Sıralı işlem teknolojisi uygulamasına rağmen hızlı matematiksel işlem yapabilme yeteneği ile ön plana çıkan başka bir işlemci türü olan DSP'ler de MPC algoritmasının gerçeklenmesinde nadiren kullanılmışlardır (Lu ve diğ. 2014).

1985 yılında Xilinx firmasının kurucuları Ross Freeman ve Bernard Vonderschmitt tarafından geliştirilen ilk FPGA olan XC2064 tanıtıldıktan sonra FPGA'ların kullanımı her geçen gün artmıştır. Geleneksel mikro işlemci tabanlı sistemlerin sıralı çalışmalarına oranla FPGA'lar üzerinde gerçeklenebilecek paralel çalışma sistemi, bir işlemin sonuçlandırılması için geçen sürede belirgin bir azalmaya yol açmaktadır. Bu özellik şu basit örnek ile açıklanabilir: (2+3)×(4+5) işlemini düşünelim. Geleneksel mikro işlemci tabanlı sistemler bu işlemi üç adımda gerçekleştirebilir: İlk adımda (2+3) işlemi yapılıp sonucu bir bellek ortamına saklanır. İkinci adımda (4+5) işlemi yapılarak sonucu başka bir bellek ortamına saklanır. Son adımda toplama işlemlerinin sonuçları bulundukları bellek ortamlarından alınarak çarpılır ve bu şekilde işlem sonucuna ulaşılmış olur. Hâlbuki uygun bir kodlama ile aynı işlem FPGA üzerinde iki adımda gerçekleştirilebilir: İlk adımda iki ayrı toplama elemanı (2+3) ve (4+5) işlemlerini aynı zaman dilimi içerisinde gerçekleştirir. İkinci işlem adımında ise toplama

elemanlarının sonuçları bir çarpma elemanı tarafından çarpılarak işlem sonucuna ulaşılır. Bu şekilde paralel işlem yapabilme yeteneği ile mevcut gömülü sistemler içinde en hızlı işlem yapabilen FPGA'lar, diğer tüm akademik ve teknolojik araştırma çalışmalarında artan oranda kullanılmasının yanı sıra, MPC algoritmalarının endüstriyel ortamlara uygulanabilmesi üzerindeki yetersizlikleri aşmada da önemli bir seçenek oluşturmaktadır. Buna rağmen literatürde FPGA'lar kullanılarak gerçekleştirilen az sayıdaki MPC uygulamalarına son 10 yıldan bu yana rastlanmaktadır.

FPGA'ların kullanıldığı ilk MPC uygulamalarında, MPC algoritmasını uygulayan temel eleman bir mikro işlemciyken, FPGA özellikle matris işlemlerini gerçekleştirmek için kullanılan bir yardımcı işlemci olarak işlev görmüştür. Örneğin 2006 yılında yapılan bir çalışmada 16-bit mikroişlemcinin yönetiminde gerçeklenen bir MPC algoritmasının ihtiyaç duyduğu matris işlemlerini gerçekleştirmek üzere Xilinx firmasının ürettiği Virtex-4 XC4VLS25 tip FPGA kullanılmış ve bu sayede etkin bir çözümleme performansı, daha az bellek kullanımı ve daha az güç tüketimi sağlandığı belirtilmiştir (Bleris ve diğ. 2006). Bu tip yapılar daha sonraları nadiren kullanılmıştır. 2012 yılında yapılan bir çalışmada bir servo motorun kontrolünü sağlamak üzere Xilinx Virtex-4 tip FPGA üzerinde donanımsal olarak gerçeklenen bir mikro işlemci MPC algoritmasını gerçekleştiren ana eleman görevini üstenirken diğer işlemler için aynı FPGA üzerinde bir yardımcı işlemci oluşturulmuştur (Yang ve diğ. 2012). İlerleyen süreçte, doğrusal MPC algoritmasının ihtiyaç duyduğu QP çözücüsünün FPGA ile gerçeklenmesinin ardından tüm MPC algoritmasının FPGA üzerinde gerçeklendiği çalışmalara rastlanmaktadır. Bu çalışmaların büyük bir kısmında kontrol edilecek sistem olarak MATLAB-Simulink üzerinde oluşturulmuş benzetimler kullanılmıştır. 2006 yılında yapılan bir çalışmada Xilinx Spartan-3L (XC3S1500L-4-fg320) FPGA yongası üzerinde tüm MPC algoritması gerçeklenmiş ve MATLAB-Simulink üzerinde oluşturulan bir doğrusal sistem benzetimi üzerinde test edilmiştir (Ling ve diğ. 2006). 2008 yılında yapılan diğer bir çalışmada MPC algoritması Xilinx Virtex-2 üzerinde paralel ve sıralı olarak iki farklı şekilde gerçeklenmiş ve yine MATLAB-Simulink üzerinde oluşturulan bir doğrusal sistem üzerinde test edilmiştir. Elde edilen sonuçlar MPC algoritmasının geleneksel mikro işlemcilerde olduğu gibi sıralı olarak işlenmesinin paralel

işlenmesine oranla 2 kattan daha fazla işlem süresine ihtiyaç duyduğunu göstermiştir (Ling ve diğ. 2008). Son zamanlarda doğrusal MPC algoritmasının tamamının FPGA üzerinde gerçeklendiği ve elde edilen yapı ile gerçek-zamanlı sistemlerin kontrol edildiği çalışmalar bulunmaktadır. 2008 yılında yapılan bir çalışmada eşzamanlı bir elektrik motorunun dönüş hızını kontrol etmek üzere Xilinx Spartan 3 XC3S400PQ208 FPGA üzerinde MPC algoritması gerçeklenmiştir (Naouar ve diğ. 2008). Bir diğer çalışmada ise dinamik modeli deneysel çalışma yoluyla elde edilen bir atomik kuvvet mikroskobu (atomic force microscope) sisteminin kontrolünü sağlamak üzere Xilinx Virtex-6 ve Spartan-3 tip FPGA'lar üzerinde hızlı gradyan (fast gradient) MPC yöntemi gerçeklenmiştir. Sabit-nokta (Fixed-point) sayı mimarisinin kullanıldığı bu iki yapı, kaynak kullanımı, çalışma hızı vb. gibi açılardan birbirleri ile karşılaştırılmış ve elde edilen sonuçlar paylaşılmıştır (Jerez ve diğ. 2012).

MPC algoritmasının gerçeklenmesini sağlamak üzere FPGA'ların programlanması için HandelC (Ling ve diğ. 2006; Ling ve diğ. 2008; Lau ve diğ. 2009; Vouzis ve diğ. 2009), Verilog Hardware Description Language (Verilog HDL) (Bleris ve diğ. 2006;Yang ve diğ. 2012) ve Very High Speed Integrated Circuit Hardware Description Language (VHDL) (Jerez ve diğ. 2013) kodlama teknikleri uygulanmıştır.

2. PROBLEM TANIMI

Bu bölümde önce doğrusal ve daha sonra doğrusal SISO sistemler için MPC açısından genel kontrol problemi tanıtılacaktır.

2.1. Doğrusal Sistemler için MPC Problemi

SISO doğrusal bir sistem

$$\begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} c_1 \\ \vdots \\ c_N \end{bmatrix}^T \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} \quad (2.1)$$

şeklinde veya

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix}, \mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix} \text{ ve } \mathbf{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_N \end{bmatrix}^T \quad (2.2)$$

olmak üzere

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t)$$

$$y(t) = \mathbf{c}\mathbf{x}(t) \quad (2.3)$$

şeklinde gösterilebilir. Burada t sürekli-zaman indeksidir ve durum değişkenleri $\mathbf{x}(t) \in \mathbb{R}^N$, giriş işareti $u(t) \in \mathbb{R}$ ve çıkış işareti de $y(t) \in \mathbb{R}$ şeklindedir. Sistem matrisleri ise $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{b} \in \mathbb{R}^{N\times 1}$ ve $\mathbf{c} \in \mathbb{R}^{1\times N}$ şeklindedir.

Kontrol yöntemi ayrık-zamanlı olduğundan böyle bir sürekli-zamanlı işaretlerin belli bir örnekleme periyodu (T_s) ile örneklenmesi gerekir. Örneklenmiş işaretler artık,

$$\tilde{y}(t)|_{t=(n+h)T_s} = \tilde{y}((n+h)T_s) = \tilde{y}[n+h]$$

$$\mathbf{x}(t)|_{t=(n+h)T_s} = \mathbf{x}((n+h)T_s) = \mathbf{x}[n+h] \qquad (2.4)$$

$$u(t)|_{t=(n+h)T_s} = u((n+h)T_s) = u[n+h]$$

şeklinde ayrık-zamanlı işaretler olarak kullanılabilir. Bu noktada, $\tilde{y}(t)$ şeklindeki bir referans işaretinin kısa vadeli bir gelecekte (kestirim ufku $t = (n+h)T_s,\ h = 1,2,\dots,H$) alacağı değerlerin $(\tilde{y}((n+h)T_s),\ h = 1,2,\dots,H)$ bilindiğini varsayalım. Bu durumda genel kontrol problemi, sistemdeki durum, giriş ve giriş değişim hızının,

$$\mathbf{x}_{min} \le \mathbf{x}((n+h)T_s) \le \mathbf{x}_{max}, \qquad h = 1,2,\dots,H$$

$$u_{min} \le u((n)T_s) \le u_{max} \qquad (2.5)$$

$$|u(nT_s) - u((n-1)T_s)| \le \Delta u_{max}$$

şeklinde verilen belirli aralıklar arasında kalma kısıtlarını sağlamaları şartıyla, sistemin çıkış işaretinin $(y(t))$ referans işaretini $(\tilde{y}(t))$ olabildiğince yakından takip edebilmesini sağlayacak şekilde uygun bir kontrol işaretinin $(u[n] = u(nT_s))$ bulunması olarak tanımlanır.

MPC problemini bu şekilde çözmek için belirli bir giriş işaretine karşı sistemin kestirim ufku boyunca üreteceği çıkışları bulabilmek gerekmektedir. Sistemin dinamik modeli kullanılarak, sistemin tüm durumlarının kestirimleri $\left(\hat{\mathbf{x}}\left((n+h)T_s\right), h = 1,2,\dots,H\right)$ ve çıkışlarının kestirimleri $\left(\hat{y}\left((n+h)T_s\right), h = 1,2,\dots,H\right)$ Doğrusal Olmayan ayrık-zamanlı sistemlerde olduğu gibi bulunabilir. Bunun için öncelikle sürekli-zamanlı sistemin ayrıklaştırılmış modelini oluşturmak gerekir. Denklem (2.3) ile verilen sürekli-zamanlı sistemin ayrıklaştırılmış modeli,

$$\hat{\mathbf{x}}[n+1] = \hat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{b}}u[n]$$

$$\hat{y}[n] = \hat{\mathbf{c}}\hat{\mathbf{x}}[n] \qquad (2.6)$$

şeklindedir. Burada $\widehat{\mathbf{A}}, \hat{\mathbf{b}}$ ve $\hat{\mathbf{c}}$ matrisleri, sistem matrislerinin Zero Order Hold (ZOH) eşdeğeri alınarak ayrıklaştırılmış halidir ve

$$\widehat{\mathbf{A}} = e^{\mathbf{A}T_S}$$

$$\hat{\mathbf{b}} = \boldsymbol{A}^{-1}(e^{\mathbf{A}T_S} - \mathbf{I})\mathbf{b} \qquad (2.7)$$

$$\hat{\mathbf{c}} = \mathbf{c}$$

şeklindedir. (2.6)'daki $\hat{\mathbf{x}}[n+1]$ ifadesi, $\hat{y}[n]$ ifadesinde yerine koyulursa,

$$\hat{y}[n+1] = \hat{\mathbf{c}}\hat{\mathbf{x}}[n+1] = \hat{\mathbf{c}}\widehat{\mathbf{A}}\hat{\mathbf{x}}[n+1] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n+1] \qquad (2.8)$$

elde edilir. Buradan hareketle sistem çıkışının kestirim ufku boyunca ilerleyen örnekleme adımlarındaki kestirimleri [n+1] örnekleme adımı için,

$$\begin{aligned} \hat{y}[n+1] &= \hat{\mathbf{c}}\hat{\mathbf{x}}[n+1] \\ &= \hat{\mathbf{c}}\big(\widehat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{b}}u[n]\big) \\ &= \hat{\mathbf{c}}\widehat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n] \end{aligned} \qquad (2.9)$$

[n+2] örnekleme adımı için,

$$\begin{aligned} \hat{y}[n+2] &= \hat{\mathbf{c}}\hat{\mathbf{x}}[n+2] \\ &= \hat{\mathbf{c}}\big(\widehat{\mathbf{A}}\hat{\mathbf{x}}[n+1] + \hat{\mathbf{b}}u[n+1]\big) \\ &= \hat{\mathbf{c}}\big(\widehat{\mathbf{A}}\big(\widehat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{b}}u[n]\big) + \hat{\mathbf{b}}u[n+1]\big) \\ &= \hat{\mathbf{c}}\big(\widehat{\mathbf{A}}\widehat{\mathbf{A}}\hat{\mathbf{x}}[n] + \widehat{\mathbf{A}}\hat{\mathbf{b}}u[n] + \hat{\mathbf{b}}u[n+1]\big) \\ &= \hat{\mathbf{c}}\widehat{\mathbf{A}}\widehat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{c}}\widehat{\mathbf{A}}\hat{\mathbf{b}}u[n] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n+1] \\ &= \hat{\mathbf{c}}\widehat{\mathbf{A}}^{\mathbf{2}}\hat{\mathbf{x}}[n] + \hat{\mathbf{c}}\widehat{\mathbf{A}}\hat{\mathbf{b}}u[n] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n+1] \end{aligned} \qquad (2.10)$$

[n+3] örnekleme adımı için,

$$
\begin{aligned}
\hat{y}[n+3] &= \hat{\mathbf{c}}\hat{\mathbf{x}}[n+3] \\
&= \hat{\mathbf{c}}\big(\hat{\mathbf{A}}\hat{\mathbf{x}}[n+2] + \hat{\mathbf{b}}u[n+2]\big) \\
&= \hat{\mathbf{c}}\big(\hat{\mathbf{A}}(\hat{\mathbf{A}}\hat{\mathbf{x}}[n+1] + \hat{\mathbf{b}}u[n+1]) + \hat{\mathbf{b}}u[n+2]\big) \\
&= \hat{\mathbf{c}}\big(\hat{\mathbf{A}}(\hat{\mathbf{A}}(\hat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{b}}u[n]) + \hat{\mathbf{b}}u[n+1]) + \hat{\mathbf{b}}u[n+2]\big) \\
&= \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{A}}\hat{\mathbf{A}}\hat{\mathbf{x}}[n] + \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{A}}\hat{\mathbf{b}}u[n] + \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{b}}u[n+1] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n+2] \\
&= \hat{\mathbf{c}}\hat{\mathbf{A}}^{\mathbf{3}}\hat{\mathbf{x}}[n] + \hat{\mathbf{c}}\hat{\mathbf{A}}^{\mathbf{2}}\hat{\mathbf{b}}u[n] + \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{b}}u[n+1] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n+2]
\end{aligned}
\tag{2.11}
$$

ve herhangi bir [n+h] örnekleme adımı için,

$$
\begin{aligned}
\hat{y}[n+h] &= \hat{\mathbf{c}}\hat{\mathbf{A}}^{h}\hat{\mathbf{x}}[n] + \hat{\mathbf{c}}\hat{\mathbf{A}}^{h-1}\hat{\mathbf{b}}u[n] + \cdots + \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{b}}u[n+h-2] + \hat{\mathbf{c}}\hat{\mathbf{b}}u[n+h-1] \\
&= \hat{\mathbf{c}}\hat{\mathbf{A}}^{h}\hat{\mathbf{x}}[n] + [\hat{\mathbf{c}}\hat{\mathbf{b}} \;\; \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{b}} \;\; \dots \;\; \hat{\mathbf{c}}\hat{\mathbf{A}}^{h-1}\hat{\mathbf{b}}]\begin{bmatrix} u[n+h-1] \\ u[n+h-2] \\ \vdots \\ u[n] \end{bmatrix}
\end{aligned}
\tag{2.12}
$$

şeklinde elde edilir. Bu kestirimler daha sade biçimde,

$$
\widehat{\mathbf{Y}}_n = \begin{bmatrix} \hat{y}[n+1] \\ \hat{y}[n+2] \\ \vdots \\ \hat{y}[n+H] \end{bmatrix} = \mathbf{Z}\hat{\mathbf{x}}[n] + \mathbf{M}\mathbf{u} \tag{2.13}
$$

şeklinde yazılabilir. Burada,

$$
\mathbf{Z} = \begin{bmatrix} \hat{\mathbf{c}}\hat{\mathbf{A}}^{1} \\ \hat{\mathbf{c}}\hat{\mathbf{A}}^{2} \\ \vdots \\ \hat{\mathbf{c}}\hat{\mathbf{A}}^{h} \end{bmatrix} \tag{2.14}
$$

ve

$$
\mathbf{M} = \begin{bmatrix} \hat{\mathbf{c}}\hat{\mathbf{b}} & 0 & \cdots & 0 \\ \hat{\mathbf{c}}\hat{\mathbf{A}}\hat{\mathbf{b}} & \hat{\mathbf{c}}\hat{\mathbf{b}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\mathbf{c}}\hat{\mathbf{A}}^{h-1}\hat{\mathbf{b}} & \hat{\mathbf{c}}\hat{\mathbf{A}}^{h-2}\hat{\mathbf{b}} & \cdots & \hat{\mathbf{c}}\hat{\mathbf{b}} \end{bmatrix} \tag{2.15}
$$

şeklindedir. Ayrıca $\hat{y}[n+h]$ denklemi kestirime başlanan örnekle adımındaki giriş işareti ($u[n]$) ile birlikte daha sonraki örnekleme adımlarındaki giriş işaretlerini ($u[n+h], h = 1,2,\dots,H$) de kapsamakta olduğu Denklem (2.12)'de açıkça görülmektedir. Bunun yanında işaretlerin değerleri bilinmediğinden bu değerlerinin ne olduğuna karar verilerek kestirim işlemine devam edilmelidir. Burada tüm kestirim ufku boyunca ($h = 1,2,\dots,H$) $u[n+h] = u[n]$ olarak kabul edildiğinden,

$$\mathbf{u} = \begin{bmatrix} u[n+h-1] \\ u[n+h-2] \\ \vdots \\ u[n] \end{bmatrix} = \begin{bmatrix} u[n] \\ u[n] \\ \vdots \\ u[n] \end{bmatrix}_{H\times 1} \qquad (2.16)$$

şeklindedir. Bunların yanı sıra kestirim ufku içindeki referans işareti de,

$$\tilde{\mathbf{Y}}_n = \begin{bmatrix} \tilde{y}[n+1] \\ \tilde{y}[n+2] \\ \vdots \\ \tilde{y}[n+H] \end{bmatrix} \qquad (2.17)$$

şeklinde gösterilebilir. Bu durumda amaç fonksiyonu

$$\begin{aligned} f(u) &= \sum_{h=1}^{H} (\tilde{y}[n+h] + \hat{y}[n+h])^2 + \lambda(u[n] - u[n-1])^2 \\ &= (\tilde{\mathbf{Y}} - \hat{\mathbf{Y}})^T (\tilde{\mathbf{Y}} - \hat{\mathbf{Y}}) + \lambda u^2 \mathbf{L} - 2\lambda u[n]u[n-1] + \lambda u^2[n-1] \\ &= (\tilde{\mathbf{Y}} - \mathbf{Z}\hat{\mathbf{x}}[n] - \mathbf{M}\mathbf{u})^T (\tilde{\mathbf{Y}} - \mathbf{Z}\hat{\mathbf{x}}[n] - \mathbf{M}\mathbf{u}) + \lambda u^2 \mathbf{L} - 2\lambda u[n]u[n-1] + \lambda u^2[n-1] \\ &= \mathbf{u}^T(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\mathbf{u} - \left(2(\tilde{\mathbf{Y}} - \mathbf{Z}\hat{\mathbf{x}}[n])^T\mathbf{M} + 2[\lambda u[n] \quad 0 \quad \dots \quad 0]\right)\mathbf{u} + S \end{aligned} \qquad (2.18)$$

şeklinde gösterilebilir. Burada $S = (\tilde{\mathbf{Y}} - \mathbf{Z}\hat{\mathbf{x}}[n])^T(\tilde{\mathbf{Y}} - \mathbf{Z}\hat{\mathbf{x}}[n]) + \lambda u^2[n-1]$ şeklindedir ve bu terim kontrol değişkenine bağlı değildir. Ayrıca λ, giriş işaretindeki ani değişimleri cezalandıran bir parametredir ve bir SISO sistem için **L**=2 değerindedir. (2.18)'den anlaşılacağı üzere, amaç fonksiyonu, kestirim izleme hatası olarak adlandırılan ve kestirim ufku içerisinde referans işaretinin ($\tilde{y}$) alacağı değerler ile sistem çıkışlarına ait kestirimler ($\hat{y}$) arasındaki farkın kareleri toplamına, giriş işareti üzerindeki değişimin karelerinin toplamının eklenmesi ile

elde edilmiştir. Sisteme uygulanacak uygun bir kontrol işareti elde edebilmek için bu amaç fonksiyonunun, verilen kısıtlamalara uymak koşulu ile giriş işaretine göre minimize edilmesi gerekir. Bu bilgiler doğrultusunda MPC problemi

$$\min_{u} f(u) = \sum_{h=1}^{H} (\tilde{y}[n+h] + \hat{y}[n+h])^2 + \lambda(u[n] - u[n-1])^2$$

$$= \mathbf{u}^T(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\mathbf{u} - \left(2\left(\tilde{\mathbf{Y}} - \mathbf{Z}\hat{\mathbf{x}}[n]\right)^T\mathbf{M} + 2[\lambda u[n] \quad 0 \quad \dots \quad 0]\right)\mathbf{u} + S \qquad (2.19)$$

Kısıtlar: $\mathbf{x}_{min} \leq \hat{\mathbf{x}}[n+h] \leq \mathbf{x}_{max}, \qquad h = 1,2,\dots,H$

$$u_{min} \leq u[n] \leq u_{max}$$

$$|u[n] - u[n-1]| \leq \Delta u_{max}$$

şeklinde ifade edilebilir.

2.2. Doğrusal Olmayan Sistemler için MPC Problemi

SISO Doğrusal Olmayan sürekli-zamanlı bir sistem,

$$\begin{bmatrix} \dot{x}_1(t) \\ \vdots \\ \dot{x}_N(t) \end{bmatrix} = \begin{bmatrix} f_1(x_1(t),\dots x_N(t), u(t)) \\ \vdots \\ f_N(x_1(t),\dots x_N(t), u(t)) \end{bmatrix} \qquad (2.20)$$

$$y(t) = g(x_1(t),\dots x_N(t))$$

şeklinde gösterilebilir. Aynı denklemlerin daha sade gösterimi,

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), u(t)) \qquad (2.21)$$

$$y(t) = g(\mathbf{x}(t))$$

şeklindedir. Burada t sürekli-zaman indeksi olup durum değişkenleri $\mathbf{x}(t) \in \mathbb{R}^N$, giriş işareti $u(t) \in \mathbb{R}$ ve çıkış işareti de $y(t) \in \mathbb{R}$ şeklindedir. Sistem fonksiyonları da $\mathbf{f}$: $\mathbb{R}^N \times \mathbb{R} \to \mathbb{R}$ ve g: $\mathbb{R}^N \to \mathbb{R}$ şeklindedir.

Kontrol yöntemi ayrık-zamanlı olduğundan sürekli-zamanlı işaretlerin belli bir örnekleme periyodu (T_s) ile örneklenmesi gerekir. Örneklenmiş işaretler artık,

$$\tilde{y}(t)|_{t=(n+h)T_s} = \tilde{y}((n+h)T_s) = \tilde{y}[n+h]$$

$$\mathbf{x}(t)|_{t=(n+h)T_s} = \mathbf{x}((n+h)T_s) = \mathbf{x}[n+h] \qquad (2.22)$$

$$u(t)|_{t=(n+h)T_s} = u((n+h)T_s) = u[n+h]$$

şeklinde ayrık-zamanlı işaretler olarak kullanılabilir.

Eğer $\tilde{y}(t)$ şeklindeki bir referans işaretinin kısa vadeli bir gelecekte (kestirim ufku: $t = (n+h)T_s,\ h = 1,2,\dots,H$) alacağı değerlerin ($\tilde{y}((n+h)T_s)$, $h = 1,2,\dots,H$) bilindiği varsayılırsa, genel kontrol problemi, sistemdeki durum, giriş ve giriş değişim hızının,

$$\mathbf{x}_{min} \le \mathbf{x}((n+h)T_s) \le \mathbf{x}_{max}, \qquad h = 1,2,\dots,H$$

$$u_{min} \le u((n)T_s) \le u_{max} \qquad (2.23)$$

$$|u(nT_s) - u((n-1)T_s)| \le \Delta u_{max},$$

şeklinde verilen belirli aralıklar arasında kalma kısıtlarını sağlamaları şartıyla, sistemin çıkış işaretinin ($y(t)$) referans işaretini ($\tilde{y}(t)$) olabildiğince yakından takip edebilmesini sağlayacak şekilde uygun bir kontrol işaretinin ($u[n] = u(nT_s)$) bulunması olarak tanımlanabilir.

MPC yöntemi ile bu problemi çözmek için belirli bir giriş işaretine karşı sistemin kestirim ufku boyunca üreteceği çıkışların bulunması gerekir. Sistemin dinamik modeli kullanılarak, sistemin tüm durumlarının kestirimleri ($\hat{\mathbf{x}}\big((n+h)T_s\big), h = 1,2,\dots,H$) ve çıkışlarının kestirimleri ($\hat{y}\big((n+h)T_s\big), h = 1,2,\dots,H$) Doğrusal Olmayan ayrık-zamanlı sistemlerde olduğu gibi bulunabilir. Bunun için öncelikle sürekli-zamanlı sistemin ayrıklaştırılmış modelini oluşturmak gerekir. Denklem (2.21) ile verilen Doğrusal Olmayan sürekli-zamanlı bir sistemin ayrıklaştırılmış modeli için en genel gösterim,

$$\hat{\mathbf{x}}\big((n+1)T_s\big) = \hat{\mathbf{f}}\big(\hat{\mathbf{x}}(nT_s), u(nT_s)\big)$$
$$\hat{y}(nT_s) = g\big(\hat{\mathbf{x}}(nT_s)\big) \qquad (2.24)$$

veya

$$\hat{\mathbf{x}}[n+1] = \hat{\mathbf{f}}(\hat{\mathbf{x}}[n], u[n])$$
$$\hat{y}[n] = g(\hat{\mathbf{x}}[n]) \qquad (2.25)$$

şeklindedir. Burada n ayrık-zaman indeksidir ve modelde durum değişkenleri $\hat{\mathbf{x}}[n] \in \mathbb{R}^N$, giriş işareti $u[n] \in \mathbb{R}$ ve çıkış işareti de $\hat{y}[n] \in \mathbb{R}$ şeklindedir. Ayrık-zamanlı modelin sistem fonksiyonları da $\hat{\mathbf{f}}: \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$ ve $g: \mathbb{R}^N \to \mathbb{R}^N$ şeklindedir. g fonksiyonu orijinal sisteminkiyle aynıdır. Fakat $\hat{\mathbf{f}}$ fonksiyonu örnekleme periyoduna (T_s) bağlı olup, orijinal sistemdeki $\mathbf{f}$ fonksiyonundan farklıdır.

Denklem (2.25) ile verilen ifadede sistem çıkışının bir sonraki örnekleme adımındaki değerini bulabilmek için,

$$\hat{y}[n+1] = g(\hat{\mathbf{x}}[n+1])$$
$$= g\left(\hat{\mathbf{f}}(\mathbf{x}[n], u[n])\right) \qquad (2.26)$$

şeklinde yazılabilir. Bu şekilde daha sonraki kestirimlerin de bulunması mümkündür:

$$\hat{y}[n+2] = g(\hat{\mathbf{x}}[n+2])$$

$$= g\left(\hat{\mathbf{f}}(\mathbf{x}[n+1], u[n+1])\right)$$

$$= g\left(\hat{\mathbf{f}}\left(\hat{\mathbf{f}}(\mathbf{x}[n], u[n]), u[n+1]\right)\right)$$

$$\hat{y}[n+3] = g(\hat{\mathbf{x}}[n+3]) \qquad (2.27)$$

$$= g\left(\hat{\mathbf{f}}(\mathbf{x}[n+2], u[n+2])\right)$$

$$= g\left(\hat{\mathbf{f}}\left(\hat{\mathbf{f}}(\mathbf{x}[n+1], u[n+1]), u[n+2]\right)\right)$$

$$= g\left(\hat{\mathbf{f}}\left(\hat{\mathbf{f}}\left(\hat{\mathbf{f}}(\mathbf{x}[n], u[n]), u[n+1]\right), u[n+2]\right)\right)$$

Sonuç olarak $n+h$ anındaki kestirimin genelleştirilmiş biçimi aşağıda gösterilen denklem ile bulunabilir:

$$\hat{y}[n+h] = g(\hat{\mathbf{x}}[n+h])$$

$$= g\left(\hat{\mathbf{f}}(\mathbf{x}[n+h-1], u[n+h-1])\right) \qquad (2.28)$$

$$= g\left(\hat{\mathbf{f}}\left(\hat{\mathbf{f}}(\mathbf{x}[n+h-2], u[n+h-2]), u[n+h-1]\right)\right)$$

Kestirim çıkışları ve kestirim ufkunda referans işareti vektör biçimi

$$\hat{\mathbf{Y}} = \begin{bmatrix} \hat{y}[n+1] \\ \hat{y}[n+2] \\ \vdots \\ \hat{y}[n+H] \end{bmatrix} \qquad (2.29)$$

ve

$$\tilde{\mathbf{Y}} = \begin{bmatrix} \tilde{y}[n+1] \\ \tilde{y}[n+2] \\ \vdots \\ \tilde{y}[n+H] \end{bmatrix} \qquad (2.30)$$

şeklinde gösterilebilir. Bu durumda Doğrusal Olmayan MPC problemi,

$$\min_{u} f(u) = \sum_{h=1}^{H} (\tilde{y}[n+h] + \hat{y}[n+h])^2 + \lambda(u[n] - u[n-1])^2$$

$$= \left(\tilde{\mathbf{Y}} - \hat{\mathbf{Y}}\right)^T \left(\tilde{\mathbf{Y}} - \hat{\mathbf{Y}}\right) + \lambda u^2 \mathbf{L} - 2\lambda u[n]u[n-1] + \lambda u^2[n-1] \quad (2.31)$$

Kısıtlar: $\mathbf{x}_{min} \leq \hat{\mathbf{x}}[n+h] \leq \mathbf{x}_{max}, \qquad h = 1,2,\dots,H$

$$u_{min} \leq u[n] \leq u_{max}$$

$$|u[n] - u[n-1]| \leq \Delta u_{max}$$

şeklinde ifade edilebilir.

3. LİTERATÜRDEKİ YÖNTEMLER

Bu bölümde, Bölüm 2'de ortaya koyulan problemin çözümü için, önce doğrusal model öngörülü kontrol anlatılacak ve daha sonra, bu tez çalışmasında geleneksel Doğrusal Olmayan model öngörülü kontrol olarak kabul edilen yöntem tanıtılacaktır.

3.1. Doğrusal Model Öngörülü Kontrol

Doğrusal sistemler için Denklem (2.18) ile verilen amaç fonksiyonu ($f(\mathbf{u})$) tasarım değişkenlerine ($\mathbf{u}$) göre kareseldir ve tasarım değişkenleri üzerindeki kısıtlar da doğrusaldır. Bu nedenle amaç fonksiyonunun minimize edilmesini sağlamak üzere bulunacak bir yerel çözüm aynı zamanda global bir çözüm olacaktır. Böyle bir problem QP problemi olarak ele alınıp çözülebilir, fakat bu oldukça hesaplama yükü ve zaman kaybına neden olur. Problemi biraz daha basitleştirmek için, global yada yerel minimum bulmak yerine, önce, amaç fonksiyonunu,

$$f(\mathbf{u}+\delta\mathbf{u}) < f(\mathbf{u}) \qquad (3.1)$$

olacak şekilde azaltan bir $\delta\mathbf{u}$ işareti bulunur. Daha sonra bu işaret, bir önceki örnekleme adımındaki kontrol işareti olan aday kontrol işaretine eklenir. Bu şekilde bulunacak bir kontrol işaretinin kullanılması suretiyle de yeterli bir kontrol performansı elde edilebilir (Maciejowski 2002). Zaten amaç fonksiyonu ($f(\mathbf{u})$) gibi bir QP probleminde eğer Newton yönünde ilerlenirse, global minimumu tek bir adımda bulmak da mümkündür.

Bunun yanı sıra, amaç fonksiyonunun ($f(\mathbf{u})$) tasarım değişkenlerine ($\mathbf{u}$) göre türevlerinden ibaret olan Gradyan ve Hessian matrislerinin kullanımına dayanan Gradyan temelli yaklaşım kullanılarak da problem çözülebilir. Bu yaklaşıma göre Gradyan vektörü,

$$\frac{\partial f(\mathbf{u})}{\partial \mathbf{u}} = 2(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\mathbf{u} - \left(2\mathbf{M}^T\left(\tilde{\mathbf{Y}}_n - \mathbf{Z}\hat{\mathbf{x}}[n]\right) + 2\begin{bmatrix} \lambda u[n-1] \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right) \qquad (3.2)$$

şeklindedir. Hessian matrisi ise,

$$\frac{\partial^2 f(\mathbf{u})}{\partial \mathbf{u}^2} = \frac{\partial}{\partial \mathbf{u}}\left(2(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\mathbf{u} - \left(2\mathbf{M}^T\left(\tilde{\mathbf{Y}}_n - \mathbf{Z}\hat{\mathbf{x}}[n]\right) + 2\begin{bmatrix} \lambda u[n-1] \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right) \right) \qquad (3.3)$$

$$= 2(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})$$

şeklinde olup pozitif tanımlıdır. Dolayısıyla $\delta\mathbf{u}$ değişimi

$$\delta\mathbf{u} = -\left(\frac{\partial^2 f(\mathbf{u})}{\partial \mathbf{u}^2}\right)^{-1} \frac{\partial f(\mathbf{u})}{\partial \mathbf{u}}$$

$$= -\mu\left(2(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\right)^{-1}\left(2(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\mathbf{u} - \left(2\mathbf{M}^T\left(\tilde{\mathbf{Y}}_n - \mathbf{Z}\hat{\mathbf{x}}[n]\right) + 2\begin{bmatrix} \lambda u[n-1] \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right) \right) \qquad (3.4)$$

$$= -\mu\mathbf{u} + \mu\left(2(\mathbf{M}^T\mathbf{M} + \lambda\mathbf{L})\right)^{-1}\left(\mathbf{M}^T\left(\tilde{\mathbf{Y}}_n - \mathbf{Z}\hat{\mathbf{x}}[n]\right) + \begin{bmatrix} \lambda u[n-1] \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right)$$

şeklinde Newton yönünde seçilerek, iteratif bir çalışma ile her adımda,

$$u^*[n] = u[n-1] + \mu\delta u[n] \qquad (3.5)$$

çözümü bulunur. Burada $u^*[n]$ sisteme uygulanacak olan kontrol işaretidir ve $0 < \mu \leq 1$ şeklinde tanımlanan μ skaler büyüklüğü, her adımda $|u[n] - u[n-1]| \leq \delta u_{max}, h = 1,2,\dots,H$ kısıtını sağlayacak şekilde seçilir.

3.2. Geleneksel Doğrusal Olmayan Model Öngörülü Kontrol

Kullanılan model doğrusal değilse kontrol problemi konveks karesel programlamadan, konveks olmayan programlama problemine dönüşür. Ancak bu durumda çözüm daha da zorlaşır. Dahası, model doğrusal değilse, global bir optimumun bulunma garantisi yoktur (Camacho ve Bordons 2007[a]). Buna rağmen

her örnekleme zamanında gerekli hesaplamalar gerçekleştirilmeli ve uygun bir kontrol işareti $u[n+1]$ elde edilmelidir. Bu problemi çözmek için 1999 yılında yapılan çalışmada belirtildiği gibi, kararlı bir kontrolör elde etmek için global bir optimum yerine, uygun yerel optimal çözümler yeterlidir (Scokaert ve diğ. 1999). Hatta bir yerel minimum bulmaya bile gerek yoktur ve her örnekleme adımında yapılması gereken tek şey, hedef fonksiyonda yeterli azalma sağlayan bir çözümün bulunmasıdır (Maciejowski 2002).

(Plucenio ve diğ. 2007)'deki çalışmada, kestirim ufku (H) boyunca gerçekleştirilen kestirimlerin ($\hat{y}[n+h]$), giriş işaretine ($u[n]$) göre türevlerinden oluşan bir Gradyan Matris kullanılarak problemi çözen bir yöntem önerilmiştir (Plucenio ve diğ. 2007). Buna yönteme göre doğrusal bir sistemin çıkışının, kestirim ufku H boyunca alacağı değerler Denklem (2.13)'te verilen çözüm ile elde edilebilir. Bu denklem,

$$\mathbf{D} = \mathbf{Z}\hat{\mathbf{x}}[n] = \begin{bmatrix} \hat{\mathbf{c}}\widehat{\mathbf{A}}^{1} \\ \hat{\mathbf{c}}\widehat{\mathbf{A}}^{2} \\ \vdots \\ \hat{\mathbf{c}}\widehat{\mathbf{A}}^{h}\hat{\mathbf{b}} \end{bmatrix} \hat{\mathbf{x}}[n],$$

$$\mathbf{G} = \mathbf{M} = \begin{bmatrix} \hat{\mathbf{c}}\hat{\mathbf{b}} & 0 & \cdots & 0 \\ \hat{\mathbf{c}}\widehat{\mathbf{A}}\hat{\mathbf{b}} & \hat{\mathbf{c}}\hat{\mathbf{b}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\mathbf{c}}\widehat{\mathbf{A}}^{h-1}\hat{\mathbf{b}} & \hat{\mathbf{c}}\widehat{\mathbf{A}}^{h}\hat{\mathbf{b}} & \cdots & \hat{\mathbf{c}}\hat{\mathbf{b}} \end{bmatrix} \quad (3.6)$$

ve

$$\Delta\mathbf{u} = \mathbf{u} = \begin{bmatrix} u[n+h-1] \\ u[n+h-2] \\ \vdots \\ u[n] \end{bmatrix} \quad (3.7)$$

olacak şekilde yeniden düzenlenirse

$$\widehat{\mathbf{Y}} = \mathbf{D} + \mathbf{G}\Delta\mathbf{u} \quad (3.8)$$

şeklinde daha sade bir gösterim elde edilir. Burada **D,** sistemin boştaki cevabını, **G,** sistemin basamak cevabını ve $\Delta\mathbf{u}$**,** giriş işareti üzerindeki değişimi göstermektedir.

Eğer kontrol edilecek sistem Doğrusal Olmayan bir sistem ise bu çözüm kullanılamaz. Fakat $\widehat{\mathbf{Y}}$ 'nin 1.-dereceden Taylor açılımını kullanan bir yaklaşıklığı yazılabilir. Buna göre Doğrusal Olmayan bir sistemin çıkışının kestirim ufku (H) boyunca alacağı değerler

$$\widehat{\mathbf{Y}} = \mathbf{D} + \mathbf{G}_{PNMPC}\Delta\mathbf{u} \qquad (3.9)$$

şeklinde bulunabilir. Burada $\mathbf{G}_{PNMPC}$; pratik Doğrusal Olmayan MPC (Practical Nonlinear MPC – PNMPC) kısaltması ile verilen gradyan matristir ve,

$$\mathbf{G}_{\mathrm{PNMPC}} = \begin{bmatrix} \frac{\partial \hat{y}_{n+1}}{\Delta u_n} \\ \frac{\partial \hat{y}_{n+2}}{\Delta u_n} \\ \vdots \\ \frac{\partial \hat{y}_{n+H}}{\Delta u_n} \end{bmatrix} \qquad (3.10)$$

şeklindedir. Böylece Denklem (3.8)'de bulunan $\mathbf{G}$ matrisi, $\mathbf{G}_{\mathrm{PNMPC}}$ matrisine dönüştürülmüştür. Bu matris kestirim ufku (H) boyunca hesaplanan sistem çıkışlarına ait kestirimlerin ($\widehat{\mathbf{Y}}$), giriş işareti değişimlerine ($\Delta\mathbf{u}$) göre türevlerinin oluşturduğu değerleri içermektedir. $\mathbf{G}_{\mathrm{PNMPC}}$ matrisinin elde edilmesinin ardından, (3.9) denkleminin iteratif olarak kullanılması ile $\widehat{\mathbf{Y}}$ bulunabilir. Bunun için Euler yöntemi veya Runge-Kutta yöntemi vb. yöntemler kullanılabilir. Elde edilen kestirim sonuçları, sistem çıkışının referans işaretini ($\widetilde{\mathbf{Y}}$) olabildiğince yakından takip etmesini sağlayacak şekilde optimize edilmesi gerekir. Yani,

$$\mathbf{F} = \left(\widetilde{\mathbf{Y}} - \widehat{\mathbf{Y}}\right)^T \left(\widetilde{\mathbf{Y}} - \widehat{\mathbf{Y}}\right) + \Delta\mathbf{u}^T\Delta\mathbf{u} \qquad (3.11)$$

şeklinde ifade edilebilecek bir amaç fonksiyonun minimize edilmesi gerekir. Kısıtlama yok ise analitik olarak çözülebilecek olan bu problem, kısıtlamalı durumda QP problemi olarak çözülebilir (Plucenio ve diğ. 2007). (3.11)'de verilen

amaç fonksiyonunun, giriş işaretine göre minimize edilmesi sonucunda elde edilen düzeltme teriminin ($\delta u[n]$), mevcut giriş işaretine ($u[n]$) eklenmesi ile, sisteme uygulanacak olan yeni kontrol işareti ($u^*[n]$) (3.12) denkleminde gösterildiği şekilde edilmiş olur.

$$u^*[n] = u[n] + \delta u[n] \qquad (3.12)$$

4. DOĞRUSAL OLMAYAN RUNGE-KUTTA MODEL ÖNGÖRÜLÜ KONTROL

Bu bölümde, tez çalışmasında kullanılan Runge-Kutta Model Öngörülü Kontrol yöntemi anlatılacaktır. Daha kolay anlaşılması için önce yöntemin dayandığı Gradyan temelli yaklaşım tanıtılacak ve daha sonra geri-beslemeli kontrol ve çevrim-içi parametre kestirimi (RK Parameter Estimation -RKPE) amaçlı olarak RKMPC/RKPE yapısının kullanımı tüm detaylarıyla verilecektir.

4.1. Gradyan Temelli Yaklaşım

Denklem (2.31) ile verilen amaç fonksiyonu, Gradyan temelli yaklaşım ile basit bir şekilde çözülebilmektedir. Bu yaklaşım, yerel yada global bir minimum noktası bulmaya çalışmak yerine, $\left(f(u[n]+\delta u[n]) < f(u[n])\right)$ eşitsizliğini sağlayacak bir düzeltme terimini $(\delta u[n])$ bulmayı amaçlamaktadır. Daha sonra, bu düzeltme teriminin aday kontrol işaretine eklenmesi ile yeterli bir kontrol performansı sağlayacak uygun bir kontrol işareti $(u^*[n] = u[n] + \delta u[n])$ elde edilebilmektedir (Maciejowski 2002). Bu amaçla, ikinci dereceden Taylor yaklaşıklığı kullanarak (2.31) ile verilen amaç fonksiyonu,

$$f(u[n]+\delta u[n]) \cong f(u[n]) + \left[\frac{\partial f(u[n])}{\partial u[n]}\right]^T \delta u[n] + \frac{1}{2}[\delta u[n]]\left[\frac{\partial^2 f(u[n])}{\partial u[n]^2}\right]\delta u[n] \qquad (4.1)$$

şeklinde tekrar düzenlenebilir. Burada $\frac{\partial f(u[n])}{\partial u[n]}$ ifadesi gradyan vektörü ve $\frac{\partial^2 f(u[n])}{\partial u[n]^2}$ ifadesi de Hessian matrisidir. Düzeltme terimi amaç fonksiyonunu minimize ettiğinden, amaç fonksiyonunun düzeltme terimine göre türevi alınıp sıfıra eşitlenirse,

$$\frac{\partial f(u[n]+\delta u[n])}{\partial \delta u[n]} \cong \frac{\partial f(u[n])}{\partial u[n]} + \left[\frac{\partial^2 f(u[n])}{\partial u[n]^2}\right]\delta u[n] = 0 \qquad (4.2)$$

elde edilir ve buradan düzeltme terimi

$$\delta u[n] = -\left[\frac{\partial^2 f(u[n])}{\partial u[n]^2}\right]^{-1} \frac{\partial f(u[n])}{\partial u[n]} \qquad (4.3)$$

şeklinde bulunur. İteratif bir şekilde bu hesaplamanın yapılması durumunda, Hessian matrisi pozitif tanımlı ve daha yüksek dereceden terimler ihmal edilebilecek durumdaysa, yerel minimuma karesel yakınsama sağlayan Newton yönünde hareket edilmiş olur (Nocedal ve Wright 1999; Venkataraman 2002). Dolayısıyla, (2.31) ile verilen amaç fonksiyonunun minimize edilmesini sağlamak amacıyla Gradyan temelli yaklaşımın kullanılabilmesi için, Gradyan $\left(\frac{\partial f(u[n])}{\partial u[n]}\right)$ ve Hessian $\left(\left[\frac{\partial^2 f(u[n])}{\partial u[n]^2}\right]\right)$ terimlerinin bulunması gerekir.

Gradyan vektörü,

$$\begin{aligned}\frac{\partial f(u)}{\partial u} &= \frac{\partial}{\partial u}\left(\left(\tilde{\mathbf{Y}}_n - \hat{\mathbf{Y}}_\mathbf{n}\right)^T\left(\tilde{\mathbf{Y}}_n - \hat{\mathbf{Y}}_\mathbf{n}\right) + \lambda u^2\mathbf{L} - 2\lambda u[n]u[n-1] + \lambda u^2[n-1]\right) \\ &= -2\left(\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}\right)\left(\tilde{\mathbf{Y}}_n - \hat{\mathbf{Y}}_\mathbf{n}\right) + 2\lambda\mathbf{L}u - 2\begin{bmatrix}\lambda u[n-1] \\ 0 \\ \vdots \\ 0\end{bmatrix}\end{aligned} \qquad (4.4)$$

şeklindedir ve $\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}$ matrisinin bulunması gerekir. Hessian matrisi ise,

$$\begin{aligned}\frac{\partial^2 f(u)}{\partial u^2} &= \frac{\partial}{\partial u}\left(-2\left(\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}\right)\left(\tilde{\mathbf{Y}}_n - \hat{\mathbf{Y}}_\mathbf{n}\right) + 2\lambda\mathbf{L}u - 2\begin{bmatrix}\lambda u[n-1] \\ 0 \\ \vdots \\ 0\end{bmatrix}\right) \\ &= -2\left(\frac{\partial^2 \hat{\mathbf{Y}}_n}{\partial u^2}\right)\left(\tilde{\mathbf{Y}}_n - \hat{\mathbf{Y}}_\mathbf{n}\right) + 2\left(\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}\right)^T\left(\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}\right) + 2\lambda\mathbf{L}\end{aligned} \qquad (4.5)$$

şeklindedir. Burada $\left(\frac{\partial^2 \hat{\mathbf{Y}}_n}{\partial u^2}\right)\left(\tilde{\mathbf{Y}}_n - \hat{\mathbf{Y}}_\mathbf{n}\right)$ ifadesinin çoğu zaman ihmal edilebilecek kadar küçük olması nedeniyle Hessian matrisi yaklaşık olarak ,

$$\frac{\partial^2 f(u)}{\partial u^2} \cong 2\left(\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}\right)^T\left(\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}\right) + 2\lambda\mathbf{L} \qquad (4.6)$$

şeklinde yazılabilir. Burada da $\frac{\partial \hat{\mathbf{Y}}_n}{\partial u}$ matrisi bilinmeyen durumundadır ve bu matris SISO sistemler için,

$$\frac{\partial \widehat{\mathbf{Y}}_n}{\partial u} = \begin{bmatrix} \hat{y}[n+1] \\ \hat{y}[n+2] \\ \vdots \\ \hat{y}[n+H] \end{bmatrix} \tag{4.7}$$

şeklinde bir vektör biçimindedir. Bu durumda,

$$\hat{\mathbf{e}} = \begin{bmatrix} \hat{e}[n+1] \\ \hat{e}[n+2] \\ \vdots \\ \hat{e}[n+H] \\ \sqrt{\lambda}\Delta u[n] \end{bmatrix} = \begin{bmatrix} \tilde{y}[n+1] - \hat{y}[n+1] \\ \tilde{y}[n+2] - \hat{y}[n+2] \\ \vdots \\ \tilde{y}[n+H] - \hat{y}[n+H] \\ \sqrt{\lambda}(u[n] - u[n-1]) \end{bmatrix} \tag{4.8}$$

ve

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial \hat{e}[n+1]}{\partial u[n]} \\ \dfrac{\partial \hat{e}[n+2]}{\partial u[n]} \\ \vdots \\ \dfrac{\partial \hat{e}[n+H]}{\partial u[n]} \\ \dfrac{\sqrt{\lambda}\,\partial(u[n] - u[n-1])}{\partial u[n]} \end{bmatrix} = - \begin{bmatrix} \dfrac{\partial \hat{y}[n+1]}{\partial u[n]} \\ \dfrac{\partial \hat{y}[n+2]}{\partial u[n]} \\ \vdots \\ \dfrac{\partial \hat{y}[n+H]}{\partial u[n]} \\ \sqrt{\lambda} \end{bmatrix} \tag{4.9}$$

olmak üzere (4.3) ile verilen düzeltme terimi,

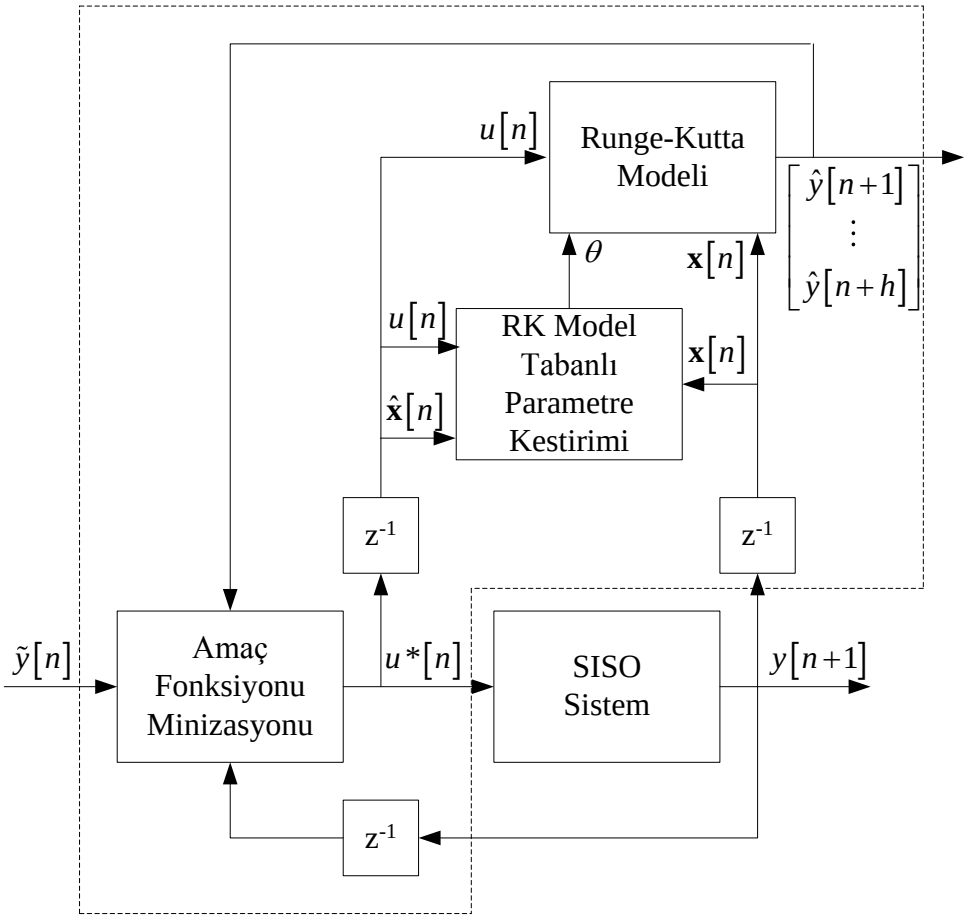
$$\delta u[n] = -(\mathbf{J}^T\mathbf{J} + \eta\mathbf{I})^{-1}\mathbf{J}^T\hat{\mathbf{e}} \tag{4.10}$$

şeklinde ifade edilebilir. Burada η parametresi $\mathbf{J}^T\mathbf{J}$ şeklinde verilen Hessian matrisinin negatif tanımlı olması durumunda, bu matrisi pozitif tanımlı yapabilecek en küçük sayıyı ifade etmektedir.

4.2. Runge-Kutta Model Öngörülü Kontrol Yapısı

Şekil 4.1'de RKMPC yapısı görülmektedir. RKMPC yönteminde kontrol edilecek Doğrusal Olmayan sürekli-zamanlı sistemin 4.-dereceden Runge-Kutta algoritması ile ayrıklaştırılmış ayrık-zamanlı modeli, Runge-Kutta Modeli olarak adlandırılmıştır. Runge-Kutta Modeli bloğu, hem bir önceki giriş işaretini hem de

durum değişkenlerini giriş olarak almaktadır. RK modelinin ürettiği durum değişkenlerine ait değerler ile sistemden doğrudan ölçülerek elde edilen durum değişkenleri değerlerinin kullanılması ile sistem dinamikleri içerisindeki parametrelerin kestirimi de gerçekleştirilebilmektedir. Bu veriler kullanılarak, Runge-Kutta Modeli bloğu içinde, amaç fonksiyonunun $(f(u))$ minimizasyonu için ihtiyaç duyulan, hem ileri yönelik sistem çıkışlarına ait kestirimler $(\hat{y}[n+h], h = 1,2,\dots,H)$, hem de düzeltme teriminin $\delta u[n]$ bulunabilmesi için gerekli Gradyan $\left(\frac{\partial f(u)}{\partial u}\right)$ ve Hessian terimleri $\left(\left[\frac{\partial^2 f(u)}{\partial u^2}\right]^{-1}\right)$ hesaplanmaktadır. Amaç Fonksiyonu Minimizasyonu bloğunda ise, RK modeli bloğunda hesaplanan veriler kullanılarak amaç fonksiyonunun minimizasyonu yapılmaktadır. Bu sayede (4.3)'de verilen düzeltme terimi $(\delta u[n])$ hesaplandıktan sonra, aday kontrol işaretine $(u[n])$ eklenerek sisteme uygulanacak kontrol işareti elde edilmektedir.



Şekil 4.1: RKMPC/RKPE yapısı

Sürekli-zamanlı sistemin ayrıklaştırılmış modeli olan Runge-Kutta Modeli 4.-dereceden Runge-Kutta algoritmasına dayanmaktadır ve bu algoritma kullanılarak, sistemin durum değişkenlerinin $(n+1)T_s$ anında alacağı değerler hesaplanabilmektedir. Buna göre, eğer $\left(\hat{\mathbf{x}}[n] = \mathbf{x}(nT_s)\right)$ olduğu kabul edilirse, herhangi bir nT_s anından başlamak üzere, daha sonraki örnekleme adımlarında RK modelinde bulunan durum değişkenlerinin alacağı değerler,

$$\begin{aligned}
\hat{x}_1[n+1] &= \hat{x}_1[n] + \frac{k_{11}[n]}{6} + \frac{k_{12}[n]}{3} + \frac{k_{13}[n]}{3} + \frac{k_{14}[n]}{6} \\
\hat{x}_2[n+1] &= \hat{x}_2[n] + \frac{k_{21}[n]}{6} + \frac{k_{22}[n]}{3} + \frac{k_{23}[n]}{3} + \frac{k_{24}[n]}{6} \\
&\vdots \\
\hat{x}_N[n+1] &= \hat{x}_N[n] + \frac{k_{N1}[n]}{6} + \frac{k_{N2}[n]}{3} + \frac{k_{N3}[n]}{3} + \frac{k_{N4}[n]}{6}
\end{aligned} \qquad (4.11)$$

şeklinde bulunabilir ve burada,

$$\begin{aligned}
k_{11}[n] &= T_s f_1(\hat{x}_1[n], \dots, \hat{x}_N[n], u[n]) \\
&\vdots \\
k_{N1}[n] &= T_s f_N(\hat{x}_1[n], \dots, \hat{x}_N[n], u[n]) \\
k_{12}[n] &= T_s f_1\left(\hat{x}_1[n] + \frac{k_{11}[n]}{2}, \dots, \hat{x}_N[n] + \frac{k_{N1}[n]}{2}, u[n]\right) \\
&\vdots \\
k_{N2}[n] &= T_s f_N\left(\hat{x}_1[n] + \frac{k_{11}[n]}{2}, \dots, \hat{x}_N[n] + \frac{k_{N1}[n]}{2}, u[n]\right) \\
k_{13}[n] &= T_s f_1\left(\hat{x}_1[n] + \frac{k_{12}[n]}{2}, \dots, \hat{x}_N[n] + \frac{k_{N2}[n]}{2}, u[n]\right) \\
&\vdots
\end{aligned} \qquad (4.12)$$

$$k_{N3}[n] = T_s f_N\left(\hat{x}_1[n] + \frac{k_{12}[n]}{2}, \dots, \hat{x}_N[n] + \frac{k_{N2}[n]}{2}, u[n]\right)$$

$$k_{14}[n] = T_s f_1(\hat{x}_1[n] + k_{13}[n], \dots, \hat{x}_N[n] + k_{N3}[n], u[n])$$

$$\vdots$$

$$k_{N4}[n] = T_s f_N(\hat{x}_1[n] + k_{13}[n], \dots, \hat{x}_N[n] + k_{N3}[n], u[n])$$

şeklindedir ve daha sade bir gösterimle,

$$\mathbf{k}[n] = \begin{bmatrix} \frac{1}{6}k_{11}[n] + \frac{1}{3}k_{12}[n] + \frac{1}{3}k_{13}[n] + \frac{1}{6}k_{14}[n] \\ \frac{1}{6}k_{21}[n] + \frac{1}{3}k_{22}[n] + \frac{1}{3}k_{23}[n] + \frac{1}{6}k_{24}[n] \\ \vdots \\ \frac{1}{6}k_{N1}[n] + \frac{1}{3}k_{N2}[n] + \frac{1}{3}k_{N3}[n] + \frac{1}{6}k_{N4}[n] \end{bmatrix} \quad (4.13)$$

şeklinde yazılabilir. Dolayısıyla (4.11) ile verilen ifade,

$$\begin{aligned} \hat{\mathbf{x}}[n+1] &= \hat{\mathbf{f}}(\hat{\mathbf{x}}[n], u[n]) \\ &= \hat{\mathbf{x}}[n] + \mathbf{k}[n] \end{aligned} \quad (4.14)$$

şeklinde gösterilebilir. Bu şekilde, RK modelinde bulunan durum değişkenlerinin kestirim ufku boyunca alacağı değerler hesaplanabilir ve bu değerler kullanılarak sistem çıkışlarına ait kestirimler de şu şekilde elde edilebilir:

$$\hat{y}[n] = g(\hat{\mathbf{x}}[n]) \quad (4.15)$$

Bu noktada, (4.10) ile verilen düzeltme terimi ifadesinin çözümü için hata vektörünün (**e**) ve Gradyan vektörünün (**J**) içinde bulunan ve kestirim çıkışlarının ($\hat{y}[n+h]$) giriş işaretine ($u[n]$) göre türevlerini ifade eden terimlerin hesaplanması gerekir. Bu terimler,

$$\frac{\partial \hat{y}[n+h]}{\partial u[n]} = \frac{\partial^T g}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\hat{\mathbf{x}}(n+h)} \frac{\partial \hat{\mathbf{x}}[n+h]}{\partial u[n]} \qquad (4.16)$$

şeklinde hesaplanır ve burada,

$$\frac{\partial \hat{\mathbf{x}}[n+h]}{\partial u[n]} = \left(\frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{x}}\bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h-1] \\ u=u[n]}} \frac{\partial \hat{\mathbf{x}}[n+h-1]}{\partial u[n]} \right) + \frac{\partial \hat{\mathbf{f}}}{\partial u}\bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h-1] \\ u=u[n]}} \qquad (4.17)$$

şeklindedir. Bu aşamadan sonra geriye sadece $\frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{x}}$, $\frac{\partial \hat{\mathbf{f}}}{\partial u}$ ve $\frac{\partial g}{\partial \mathbf{x}}$ terimlerinin bulunması kalmaktadır. (4.18) denkleminden itibaren kolaylık olması açısından zaman indeksi dikkate alınmamıştır. Kontrol edilecek sistemin ayrık-zamanlı modelini ifade eden RK modelinin sistem durum değişkenlerine göre türevi,

$$\begin{aligned} \frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{x}} &= \frac{\partial (\mathbf{x}+\mathbf{k})}{\partial \mathbf{x}} \\ &= \mathbf{I} + \frac{\partial \mathbf{k}}{\partial \mathbf{x}} \end{aligned} \qquad (4.18)$$

şeklindedir ve (4.13) ifadesi,

$$\mathbf{k}_1 = \begin{bmatrix} k_{11} \\ k_{21} \\ \vdots \\ k_{N1} \end{bmatrix} = \begin{bmatrix} T_s f_1(\mathbf{x}, u) \\ T_s f_2(\mathbf{x}, u) \\ \vdots \\ T_s f_N(\mathbf{x}, u) \end{bmatrix} = T_s \mathbf{f}(\mathbf{x}, u)$$

$$\mathbf{k}_2 = \begin{bmatrix} k_{12} \\ k_{22} \\ \vdots \\ k_{N2} \end{bmatrix} = \begin{bmatrix} T_s f_1(\mathbf{x} + 0.5\mathbf{k}_1, u) \\ T_s f_2(\mathbf{x} + 0.5\mathbf{k}_1, u) \\ \vdots \\ T_s f_N(\mathbf{x} + 0.5\mathbf{k}_1, u) \end{bmatrix} = T_s \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_1, u)$$

$$\mathbf{k}_3 = \begin{bmatrix} k_{13} \\ k_{23} \\ \vdots \\ k_{N3} \end{bmatrix} = \begin{bmatrix} T_s f_1(\mathbf{x} + 0.5\mathbf{k}_2, u) \\ T_s f_2(\mathbf{x} + 0.5\mathbf{k}_2, u) \\ \vdots \\ T_s f_N(\mathbf{x} + 0.5\mathbf{k}_2, u) \end{bmatrix} = T_s \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_2, u) \qquad (4.19)$$

$$\mathbf{k}_4 = \begin{bmatrix} k_{14} \\ k_{24} \\ \vdots \\ k_{N4} \end{bmatrix} = \begin{bmatrix} T_s f_1(\mathbf{x} + \mathbf{k}_3, u) \\ T_s f_2(\mathbf{x} + \mathbf{k}_3, u) \\ \vdots \\ T_s f_N(\mathbf{x} + \mathbf{k}_3, u) \end{bmatrix} = T_s \mathbf{f}(\mathbf{x} + \mathbf{k}_3, u)$$

şeklinde düzenlenirse

$$\frac{\partial \mathbf{k}}{\partial \mathbf{x}} = \frac{\partial\left(\frac{1}{6}\mathbf{k}_1 + \frac{1}{3}\mathbf{k}_2 + \frac{1}{3}\mathbf{k}_3 + \frac{1}{6}\mathbf{k}_4\right)}{\partial \mathbf{x}} \qquad (4.20)$$

$$= \frac{1}{6}\frac{\partial \mathbf{k}_1}{\partial \mathbf{x}} + \frac{1}{3}\frac{\partial \mathbf{k}_2}{\partial \mathbf{x}} + \frac{1}{3}\frac{\partial \mathbf{k}_3}{\partial \mathbf{x}} + \frac{1}{6}\frac{\partial \mathbf{k}_4}{\partial \mathbf{x}}$$

elde edilir. Burada,

$$\frac{\partial \mathbf{k}_1}{\partial \mathbf{x}} = \frac{\partial T_s \mathbf{f}(\mathbf{x}, u)}{\partial \mathbf{x}}$$

$$= T_s \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}$$

$$\frac{\partial \mathbf{k}_2}{\partial \mathbf{x}} = \frac{\partial T_s \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_1, u)}{\partial \mathbf{x}}$$

$$= T_s \frac{\partial \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_1, u)}{\partial \mathbf{x}}$$

$$= T_s \frac{\partial \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_1, u)}{\partial(\mathbf{x} + 0.5\mathbf{k}_1)} \frac{\partial(\mathbf{x} + 0.5\mathbf{k}_1)}{\partial \mathbf{x}}$$

$$= T_s \left(\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_1} \left(\mathbf{I} + 0.5\frac{\partial \mathbf{k}_1}{\partial \mathbf{x}}\right)\right) \qquad (4.21)$$

$$\frac{\partial \mathbf{k}_3}{\partial \mathbf{x}} = \frac{\partial T_s \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_2, u)}{\partial \mathbf{x}}$$

$$= T_s \frac{\partial \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_2, u)}{\partial \mathbf{x}}$$

$$= T_s \frac{\partial \mathbf{f}(\mathbf{x} + 0.5\mathbf{k}_2, u)}{\partial(\mathbf{x} + 0.5\mathbf{k}_2)} \frac{\partial(\mathbf{x} + 0.5\mathbf{k}_2)}{\partial \mathbf{x}}$$

$$= T_s \left(\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_2} \left(\mathbf{I} + 0.5\frac{\partial \mathbf{k}_2}{\partial \mathbf{x}}\right)\right)$$

$$\frac{\partial \mathbf{k}_4}{\partial \mathbf{x}} = \frac{\partial T_s \mathbf{f}(\mathbf{x} + \mathbf{k}_3, u)}{\partial \mathbf{x}}$$

$$= T_s \frac{\partial \mathbf{f}(\mathbf{x}+\mathbf{k}_3, u)}{\partial \mathbf{x}}$$

$$= T_s \frac{\partial \mathbf{f}(\mathbf{x}+\mathbf{k}_3, u)}{\partial (\mathbf{x}+\mathbf{k}_3)} \frac{\partial (\mathbf{x}+\mathbf{k}_3)}{\partial \mathbf{x}}$$

$$= T_s \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+\mathbf{k}_3} \left(\mathbf{I} + \frac{\partial \mathbf{k}_3}{\partial \mathbf{x}} \right) \right)$$

şeklindedir. Benzer şekilde RK modelinin giriş işaretine göre türevi,

$$\begin{aligned} \frac{\partial \hat{\mathbf{f}}}{\partial u} &= \frac{\partial (\mathbf{x}+\mathbf{k})}{\partial u} \\ &= \frac{\partial \mathbf{k}}{\partial u} \end{aligned} \qquad (4.22)$$

şeklindedir ve (4.36)'dan faydalanarak,

$$\begin{aligned} \frac{\partial \mathbf{k}}{\partial u} &= \frac{\partial \left(\frac{1}{6}\mathbf{k}_1 + \frac{1}{3}\mathbf{k}_2 + \frac{1}{3}\mathbf{k}_3 + \frac{1}{6}\mathbf{k}_4 \right)}{\partial u} \\ &= \frac{1}{6}\frac{\partial \mathbf{k}_1}{\partial u} + \frac{1}{3}\frac{\partial \mathbf{k}_2}{\partial u} + \frac{1}{3}\frac{\partial \mathbf{k}_3}{\partial u} + \frac{1}{6}\frac{\partial \mathbf{k}_4}{\partial u} \end{aligned} \qquad (4.23)$$

yazılabilir. Burada,

$$\begin{aligned} \frac{\partial \mathbf{k}_1}{\partial u} &= \frac{\partial T_s \mathbf{f}(\mathbf{x}, u)}{\partial u} \\ &= T_s \frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}} \\ \frac{\partial \mathbf{k}_2}{\partial u} &= \frac{\partial T_s \mathbf{f}(\mathbf{x}+0.5\mathbf{k}_1, u)}{\partial u} \\ &= T_s \frac{\partial \mathbf{f}(\mathbf{x}+0.5\mathbf{k}_1, u)}{\partial u} \\ &= T_s \frac{\partial \mathbf{f}(\mathbf{x}+0.5\mathbf{k}_1, u)}{\partial (\mathbf{x}+0.5\mathbf{k}_1)} \frac{\partial (\mathbf{x}+0.5\mathbf{k}_1)}{\partial u} + \frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_1} \end{aligned} \qquad (4.24)$$

$$= T_s\left(0.5\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_1}\frac{\partial \mathbf{k}_1}{\partial u}+\frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_1}\right)$$

$$\frac{\partial \mathbf{k}_3}{\partial u}=\frac{\partial T_s\mathbf{f}(\mathbf{x}+0.5\mathbf{k}_2,u)}{\partial u}$$

$$= T_s\frac{\partial \mathbf{f}(\mathbf{x}+0.5\mathbf{k}_2,u)}{\partial u}$$

$$= T_s\frac{\partial \mathbf{f}(\mathbf{x}+0.5\mathbf{k}_2,u)}{\partial(\mathbf{x}+0.5\mathbf{k}_2)}\frac{\partial(\mathbf{x}+0.5\mathbf{k}_2)}{\partial u}+\frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_2}$$

$$= T_s\left(0.5\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_2}\frac{\partial \mathbf{k}_2}{\partial u}+\frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+0.5\mathbf{k}_2}\right)$$

$$\frac{\partial \mathbf{k}_4}{\partial u}=\frac{\partial T_s\mathbf{f}(\mathbf{x}+\mathbf{k}_3,u)}{\partial u}$$

$$= T_s\frac{\partial \mathbf{f}(\mathbf{x}+\mathbf{k}_3,u)}{\partial u}$$

$$= T_s\frac{\partial \mathbf{f}(\mathbf{x}+\mathbf{k}_3,u)}{\partial(\mathbf{x}+\mathbf{k}_3)}\frac{\partial(\mathbf{x}+\mathbf{k}_3)}{\partial u}+\frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+\mathbf{k}_3}$$

$$= T_s\left(0.5\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+\mathbf{k}_3}\frac{\partial \mathbf{k}_3}{\partial u}+\frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}+\mathbf{k}_3}\right)$$

şeklindedir ve son olarak sistem çıkış fonksiyonunun durum değişkenlerine göre türevi,

$$\frac{\partial g}{\partial \mathbf{x}}=\frac{\partial g}{\partial \mathbf{x}} \qquad (4.25)$$

şeklindedir.

Bu şekilde elde edilen sistem çıkışına ait kestirim sonuçları ($\hat{y}[n+h]$, $h=1,\dots,H$) ve türev değerleri $\left(\frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{x}},\frac{\partial \hat{\mathbf{f}}}{\partial u},\frac{\partial g}{\partial \mathbf{x}}\right)$ (4.10)'da yerine koyulduğunda düzeltme terimi ($\delta u[n]$) elde edilmiş olur. Daha sonra sisteme uygulanacak kontrol işareti,

$$u^*[n]=u[n]+\mu\delta u[n] \qquad (4.26)$$

şeklinde hesaplanır. Burada $0 < \mu < 1$ şeklinde tanımlanan parametre,

$$\mu = max\left(\frac{u_{min} - u}{\delta u[n]}, \frac{u_{max} - u}{\delta u[n]}\right) \quad (4.27)$$

şeklindedir ve sisteme uygulanacak kontrol işaretinin $u_{min} - u_{max}$ aralığında kalmasını sağlamaktadır.

4.3. Parametre Kestirimi

RK modeli kullanılarak, sistem dinamikleri içinde bulunan parametrelerin (θ) çevrim-içi şekilde kestirimi de mümkündür. Bunun için önce, $(n+1)$ numaralı örnekleme adımında, Doğrusal Olmayan sistemin bir önceki örnekleme adımına ait durum değerleri $(\boldsymbol{x}[n])$ ve mevcut durum değerleri $(\boldsymbol{x}[n+1])$ ile birlikte bir önceki örnekleme adımına ait kontrol işaretinin $(u[n])$ bilindiğini varsayalım. Bu durumda (4.11) kullanılarak, sistemin durum, giriş ve parametrelerinin bir önceki değerleri ile mevcut değerleri ilişkilendirilebilir. Bu şekilde, parametrelerin (θ) kestirimini sağlamak için önce, sistemden elde edilen durum değerleri ile RK modelinin ürettiği durum değerlerinin farkından oluşan bir kestirim hatası vektörü $(\mathbf{e})$ oluşturulur:

$$\mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} x_1[n+1] - \hat{x}_1[n+1] \\ x_2[n+1] - \hat{x}_2[n+1] \\ \vdots \\ x_N[n+1] - \hat{x}_N[n+1] \end{bmatrix} = \left[\mathbf{x}[n+1] - \hat{\mathbf{x}}[n+1]\right] \quad (4.28)$$

Daha sonra bu hata vektörünün parametrelere göre türevlerinden oluşan bir Jacobian matrisi oluşturulur:

$$\mathbf{J}_\theta = \left[\frac{\partial e_1}{\partial \theta} \quad \frac{\partial e_2}{\partial \theta} \quad \cdots \quad \frac{\partial e_N}{\partial \theta}\right]^T = \left[\frac{\partial \mathbf{e}}{\partial \theta}\right] \quad (4.29)$$

Son olarak mevcut parametre değerleri,

$$\theta[n+1] = \theta[n] - \frac{\mathbf{J}_\theta^T \mathbf{e}}{\mathbf{J}_\theta^T \mathbf{J}_\theta} \quad (4.30)$$

şeklinde güncellenir. Dolayısıyla, parametre kestirimi gerçekleştirebilmek için, RK modelinde bulunan durum değişkenlerinin, kestirimi yapılacak parametrelere göre türevlerinin oluşturduğu Jacobian matrisinin elde edilmesi gerekmektedir. Bu matrisin her bir elemanı $i = 1,2,\dots,N$ olmak üzere,

$$\frac{\partial e_i}{\partial \theta} = \frac{\partial (x_i[n+1] - \hat{x}_i[n+1])}{\partial \theta} \qquad (4.31)$$

şeklindedir ve sistemden elde edilen durum değişkenlerini değerleri $(x_i[n+1])$ sabit sayılar şeklinde olduklarından aynı ifade,

$$\frac{\partial e_i}{\partial \theta} = -\frac{\partial \hat{x}_i[n+1]}{\partial \theta} \qquad (4.32)$$

şeklinde yazılabilir. Bu kısmi türevler (4.17)'den faydalanarak,

$$\frac{\partial \hat{\mathbf{x}}[n+1]}{\partial \theta} = \left(\frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{x}}\bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h-1] \\ u=u[n] \\ \theta=\theta[n]}} \frac{\partial \hat{\mathbf{x}}[n+h-1]}{\partial \theta} \right) + \frac{\partial \hat{\mathbf{f}}}{\partial \theta}\bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h-1] \\ u=u[n] \\ \theta=\theta[n]}} \qquad (4.33)$$

şeklinde hesaplanabilir. Parametre kestirimi için sadece bir örnekleme adımı sonrası için RK modeli kullanılacağından (4.33) denkleminde,

$$\frac{\partial \hat{\mathbf{x}}[n+h-1]}{\partial \theta} = 0 \qquad (4.34)$$

olacaktır ve dolayısıyla Jacobian matrisi,

$$\mathbf{J}_\theta = -\left[\frac{\partial \mathbf{e}}{\partial \theta}\right] = -\left[\frac{\partial \hat{\mathbf{x}}[n+1]}{\partial \theta}\right] = -\left[\frac{\partial \hat{\mathbf{f}}}{\partial \theta}\right]\bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h] \\ u=u[n] \\ \theta=\theta[n]}} \qquad (4.35)$$

şeklinde elde edilir. Burada,

$$\frac{\partial \hat{\mathbf{f}}}{\partial \theta} = \frac{\partial (\hat{\mathbf{x}} + \mathbf{k})}{\partial \theta} = \frac{\partial \mathbf{k}}{\partial \theta} \qquad (4.36)$$

şeklindedir ve

$$\frac{\partial \mathbf{k}_1}{\partial \theta} = T_s \left[\frac{\partial \mathbf{f}}{\partial \theta}\right]_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n] \\ u=u[n] \\ \theta=\theta[n]}}$$

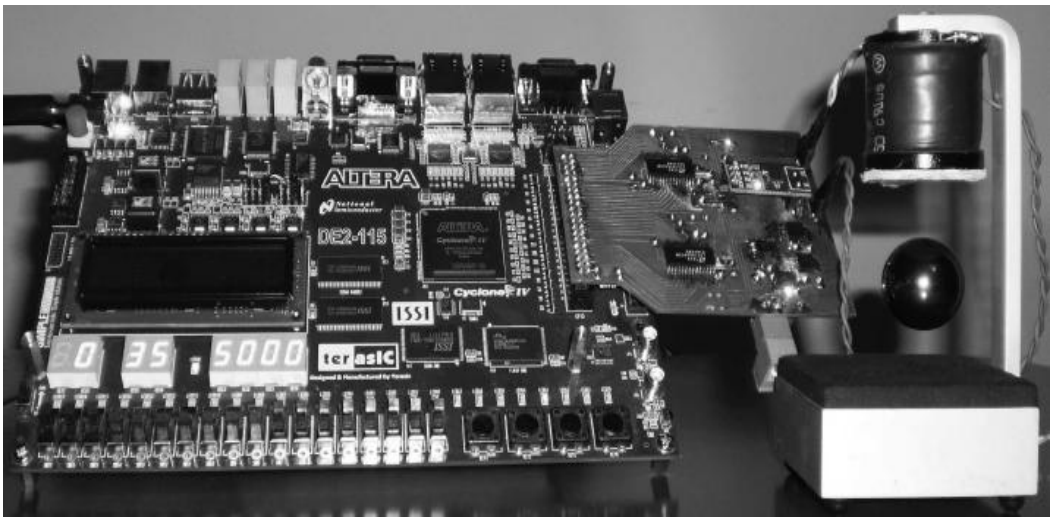
$$\frac{\partial \mathbf{k}_2}{\partial \theta} = T_s \left(\frac{\partial \mathbf{f}}{\partial \theta} + 0.5\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_1}{\partial \theta}\right)\Bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n]+0.5\mathbf{k}_1 \\ u=u[n] \\ \theta=\theta[n]}} \qquad (4.37)$$

$$\frac{\partial \mathbf{k}_3}{\partial \theta} = T_s \left(\frac{\partial \mathbf{f}}{\partial \theta} + 0.5\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_2}{\partial \theta}\right)\Bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n]+0.5\mathbf{k}_2 \\ u=u[n] \\ \theta=\theta[n]}}$$

$$\frac{\partial \mathbf{k}_4}{\partial \theta} = T_s \left(\frac{\partial \mathbf{f}}{\partial \theta} + 0.5\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_3}{\partial \theta}\right)\Bigg|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n]+\mathbf{k}_3 \\ u=u[n] \\ \theta=\theta[n]}}$$

şeklindedir. Böylece parametre kestirimi için gerekli olan türev hesaplamaları da sistemin Runge-Kutta Modeli kullanılarak gerçekleştirilmiş olur.

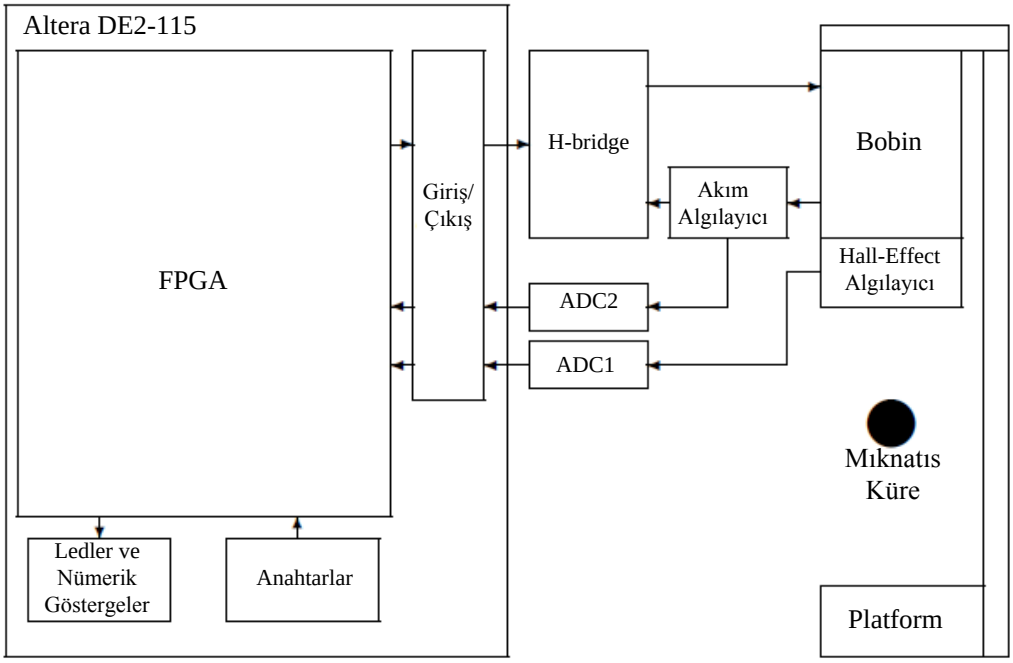
5. RKMPC YÖNTEMİNİN FPGA ÜZERİNDE GÖMÜLÜ SİSTEM ŞEKLİNDE GERÇEKLENMESİ



Şekil 5.1: RKMPC yönteminin donanımsal gerçeklemesi

Bu tez çalışmasında, Doğrusal Olmayan Runge-Kutta Model Öngörülü Kontrol (RKMPC) yöntemi bir FPGA üzerinde gerçeklenmiş ve elde edilen kontrolör, gerçek-zamanlı, deneysel amaçlı bir EMLS'nin kontrolünü sağlamak ve çevrim-içi parametre kestirimi yapmak amacıyla kullanılmıştır. Oluşturulan geribeslemeli kontrol sisteminin Şekil 5.1'de bir resmi ve Şekil 5.2'de genel yapısı görülmektedir. Buna göre, geribeslemeli kontrol sistemi,

- DE2-115 FPGA eğitim ve geliştirme seti,
- Üzerinde iki kanal Analog-Sayısal Çevirici (Analog-Digital Converter – ADC) ve H-bridge MOSFET yapısındaki bir bobin sürücü bulunan ek kart
- Kontrol edilecek sistem olarak kullanılan EMLS'den oluşmaktadır.



Şekil 5.2: Geribeslemeli kontrol sisteminin bileşenleri

Bu bölümde önce, kullanılan FPGA eğitim ve geliştirme seti ve bununla birlikte kullanılmak üzere tasarlanan ek kart tanıtılacak, ardından kontrol edilecek sistem olarak kullanılan EMLS tanıtılacak ve son olarak RKMPC/RKPE yönteminin FPGA üzerinde gerçeklenmesi ile ilgili tüm detaylar verilecektir.

5.1. DE2-115 FPGA Eğitim ve Geliştirme Seti

Bu tez çalışmasında, üzerinde CYCLONE-IV EP4CE115F29C7 tip FPGA ve çok çeşitli işlemlere yönelik olarak yerleştirilmiş, çok çeşitli birimler bulunan DE2-115 eğitim ve geliştirme seti kullanılmıştır. ALTERA firmasının ürettiği bu setin üzerinde bulunan çok çeşitli birimlerden bu tez çalışmasında kullanılanlar Şekil 5.3'te gösterilmiştir. Bu birimler şu şekilde tanıtılabilir (Altera 2015):

- **FPGA:** Setin üzerinde Cyclone-IV EP4CE115 tip bir FPGA bulunmaktadır. Bu FPGA içerisinde 114480 adet mantık elemanı (Logic Elements - LEs), 3888 Kbit gömülü bellek (Embedded Memory Bits– MBs), 18×18 bit işlem kapasitesine sahip 266 adet gömülü çarpma elemanı (Embedded Multipliers – EMs), 4 adet genel amaçlı faz kilitlemeli döngü (Phase Locked Loop – PLL)

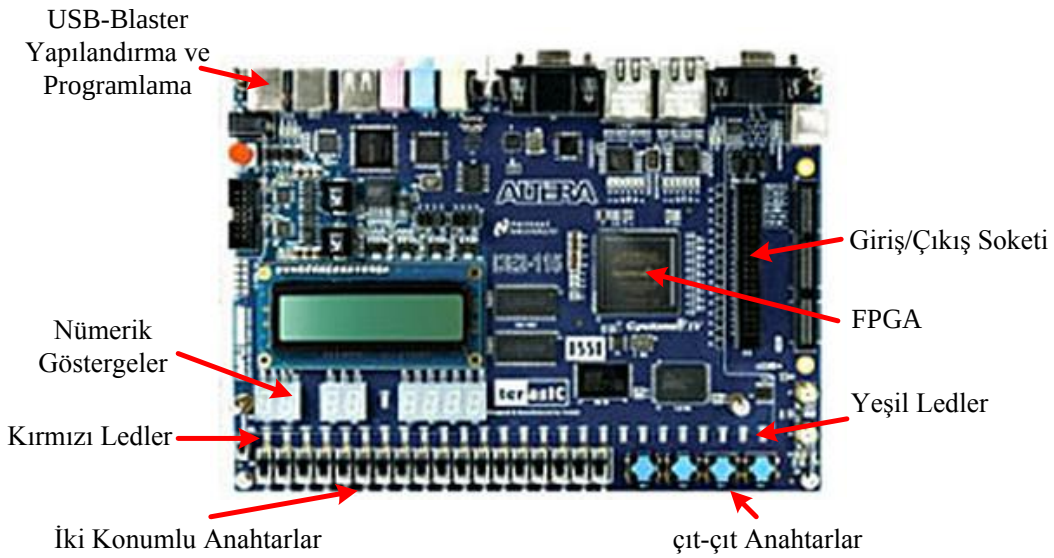
elemanı ve 528 adet genel amaçlı giriş/çıkış bacağı (General Purpose Inputs/Outputs) bulunmaktadır.

- **Çalışma Osilatörü:** Kart üzerine yerleştirilmiş durumda 50MHz frekanslı 3 adet osilatör bulunmaktadır.

- **Yapılandırma ve Programlama:** DE2-115 eğitim ve geliştirme seti üzerinde EPCS64 seri yapılandırma cihazı (Serial Configuration Device), kart üzerine yerleştirilmiş durumda USB üzerinden bilgisayar bağlantısı yapılan Blaster adlı programlama devresi bulunmaktadır. Bunun yanı sıra, müşterek test eylemi grubu (Joint Test Action Group – JTAG) kipi ve eş-zamanlı olmayan seri haberleşme (Asynchronous Serial Communication) kipi ile programlama, çevrim-içi veri haberleşmesi vb. gibi yapılandırma işlemleri yapılabilmektedir. FPGA üzerinde oluşturulacak donanımı ifade eden yazılım istenirse RAM türü veya FLASH ROM türü bellek üzerine aktarılabilmektedir.

- **Kullanıcı Arabirimleri:** Kullanıcıya sunulmuş olan 18 adet iki konumlu anahtar (switches) ve 4 adet çıt-çıt anahtar (buttons), 18 adet kırmızı ve 9 adet yeşil LED ve 8 adet 7-parçalı gösterge bulunmaktadır.

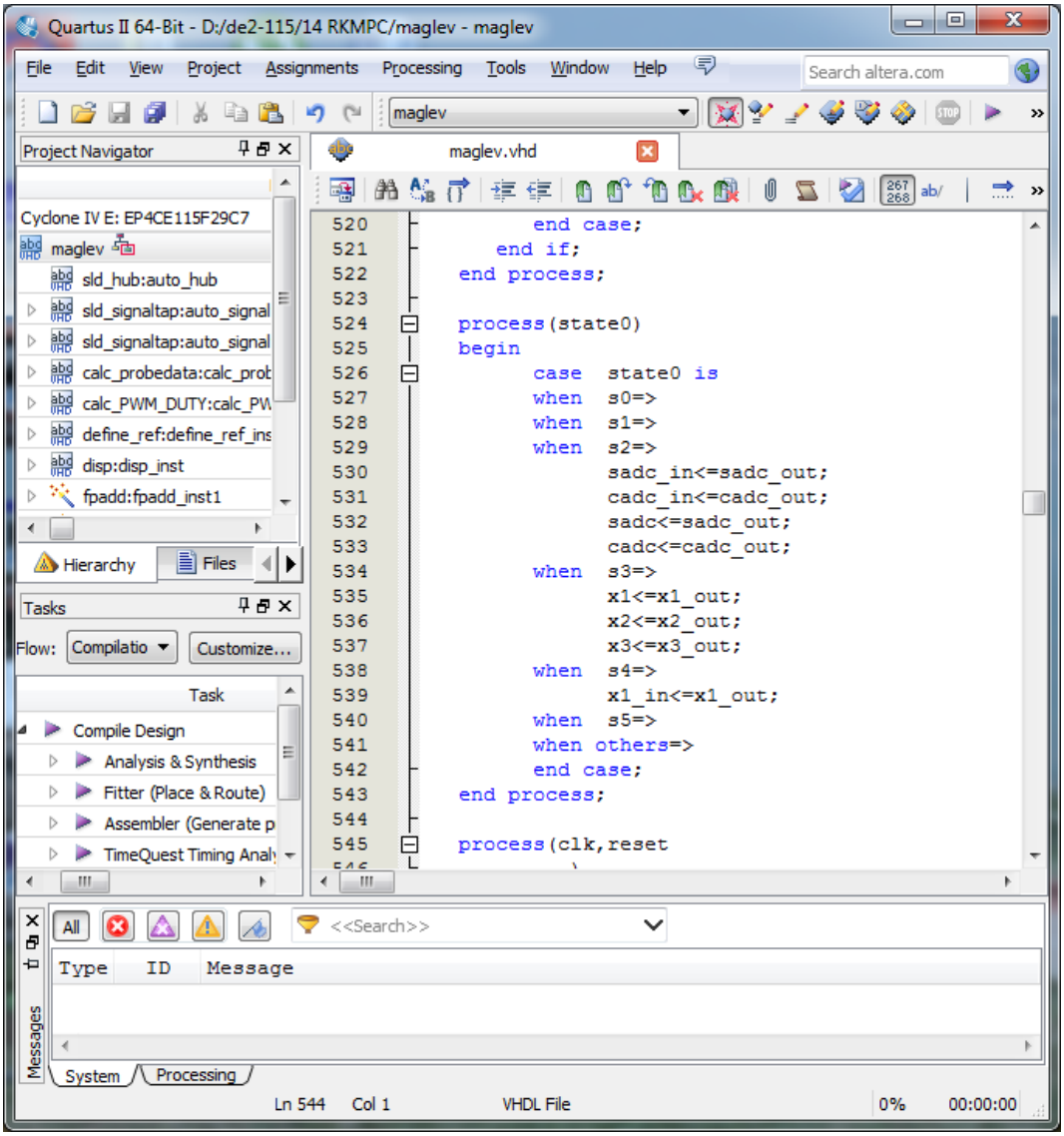
- **Giriş/Çıkış Bağlantı Noktası:** Set üzerinde bulunan 40 iğneli (pin) giriş çıkış soketi ek kart bağlantısı için kullanılmıştır.



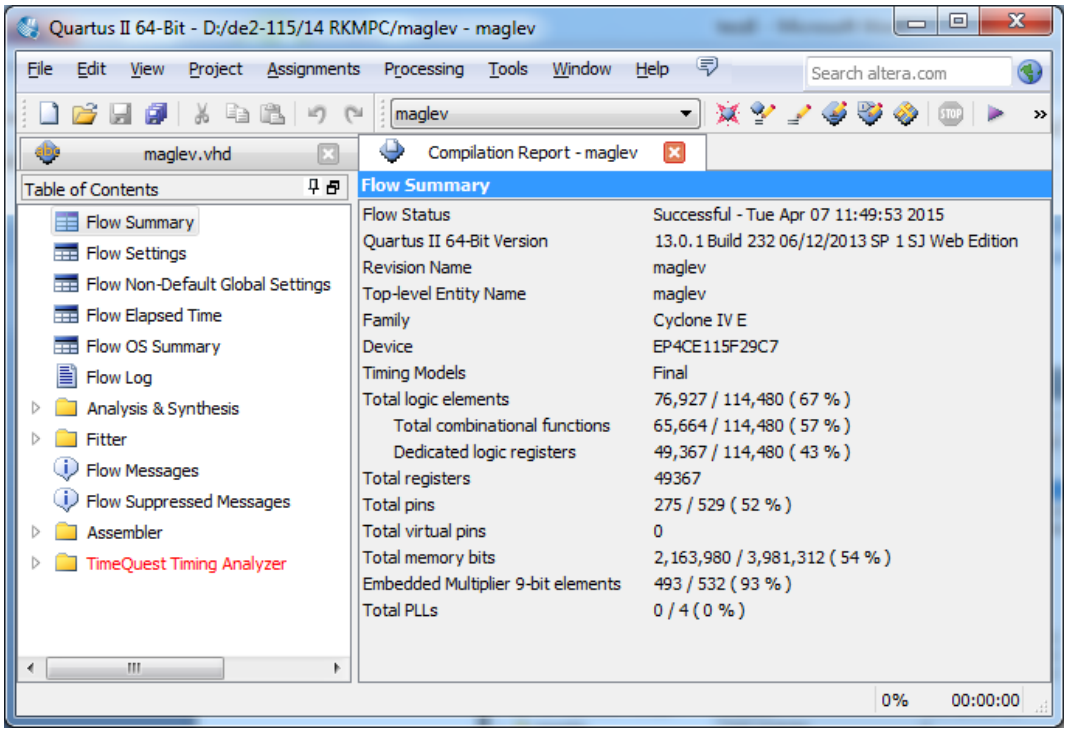
Şekil 5.3: DE2-115 eğitim ve geliştirme seti

DE2-115 seti üzerinde bir çalışma gerçekleştirmek için, FPGA'ya yüklenecek yazılım, Quartus II yazılım geliştirme aracı kullanılarak oluşturulabilmektedir. Altera firmasının Web Edition adı altında ücretsiz kullanıma sunduğu bu yazılım geliştirme aracı ile bir FPGA üzerinde istenen donanımı oluşturmak için en çok bilinen yazılım modelleri olan VHDL ve Verilog (ya da Verilog HDL) dillerini kullanarak yazılım geliştirmek mümkündür. Bu tez çalışmasında RKMPC/RKPE algoritmasının FPGA üzerinde gerçeklenebilmesi için gerekli yazılım VHDL tipi kodlama tekniği kullanılarak oluşturulmuştur. Quartus II yazılım geliştirme aracı ile bütünleşik olarak verilen ve Tools menüsü altında bulunan pek çok küçük yardımcı programlar bulunmaktadır. Bunlardan SignalTap II Logic analyzer, In-System Sources and Probes ve Megafunction Plug-in Wizard araçları bu tez çalışması sırasında kullanılmış ve bunlara ilişkin detaylar Bölüm 5.7'de anlatılmıştır. Ayrıca, MegaFunction olarak adlandırılan ve çeşitli amaçlara yönelik olarak önceden hazırlanmış olan VHDL ve Verilog türünden yazılım parçaları da Quartus II programının yüklenmesi sırasında bilgisayara aktarılmaktadır. Intellectual Property (IP) çekirdekleri olarak da adlandırılan bu tür küçük hazır FPGA yazılımları arasında, 32-bit/64-bit olarak seçilebilen ve floating point (kayan nokta)/ fixed point (sabit nokta) yapısı kullanan matematiksel işlemciler, Quartus II ile FPGA arasında iletişimi sağlamak için kullanılan iletişimciler, FPGA içindeki ve dışındaki bellek bölgelerine ulaşmak ve düzenlemek için kullanılan bellekciler sayılabilir. Şekil 5.4'de Quartus II yazılım geliştirme aracının ekran görüntüsü görülmektedir.

Quartus II üzerinde gerçekleştirilen yazılımlar, FPGA üzerinde aktarılmadan önce derleme işlemine tabi tutulmalıdır ve bu işlem Quartus II içinde bütünleşik halde bulunan derleyici ile gerçekleştirilebilmektedir. Derleme işlemi Analysis/Synthesis, Fitter, Assembler ve TimeQuest Timing Analysis adımlarından oluşmaktadır. Bu işlemler sırasında elde edilen tüm bilgiler, derleme sonrasında Processing menüsü altındaki Compilation Report seçeneği tıklanarak ekrana gelen pencerede görülebilmektedir. Bu pencerenin görüntüsü Şekil 5.5'te verilmiştir.



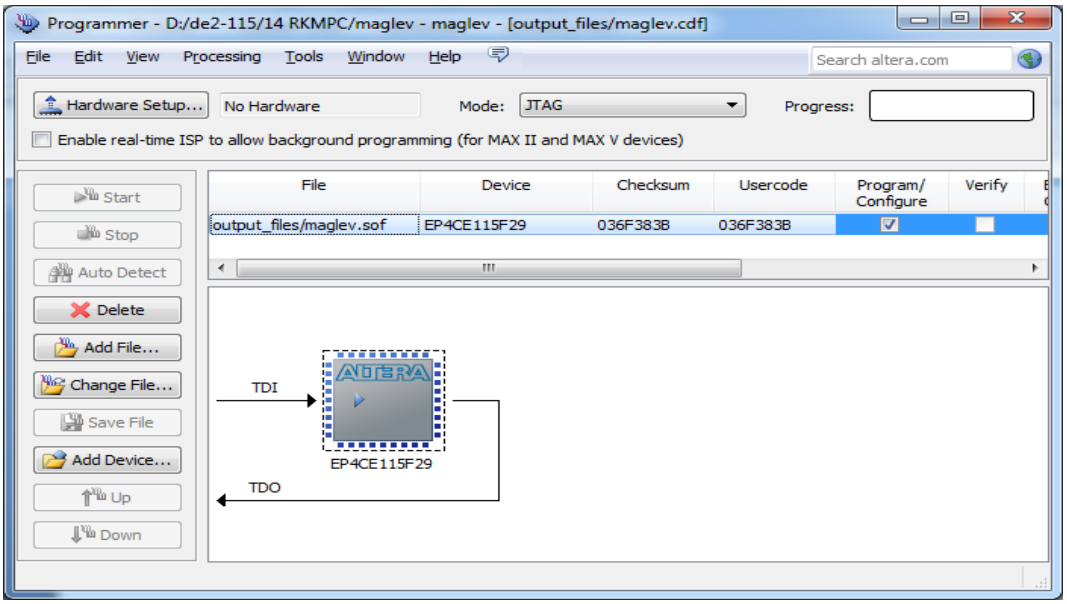
Şekil 5.4: Quartus II (v13.0sp1 64-bit) ekran görüntüsü



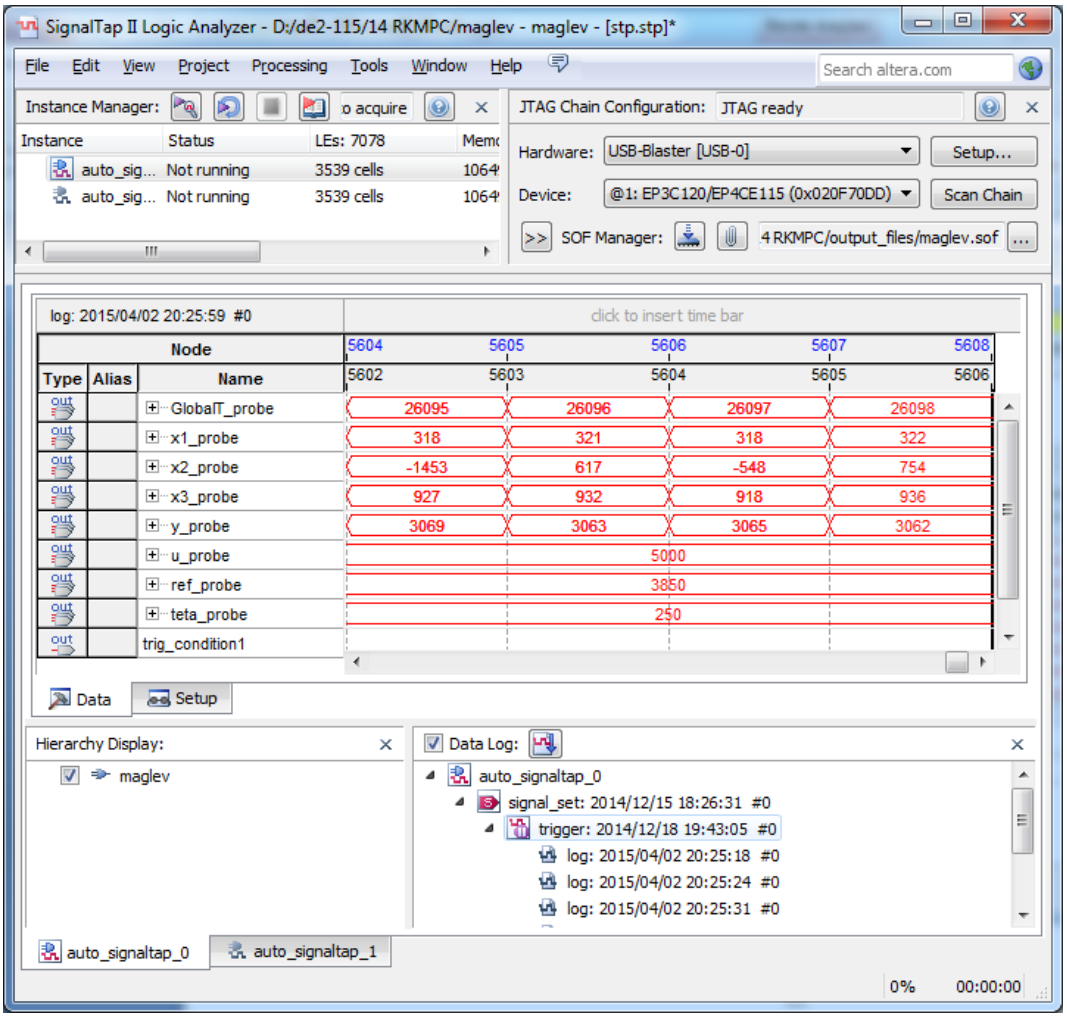
Şekil 5.5: Compilation Report penceresi ekran görüntüsü

Derleme sonrasında elde edilen FPGA kodlarının DE2-115'ya gönderimi, DE2-115 üzerinde bütünleşik halde bulunan USB-Blaster programlama devresi kullanılarak gerçekleştirilebilmektedir. Bu araç bilgisayar ile DE2-115 arasında USB fiziksel bağlantı yolu üzerinden iletişimi sağlayarak, program kodlarının FPGA'ya aktarılmasını sağlamaktadır. Derleme sonrasında elde edilen kodların USB-Blaster yapısı üzerinden FPGA'ya gönderilebilmesi için Quartus II içinde bütünleşik olarak bulunan Programmer aracı kullanılmaktadır. Bu aracın ekran görüntüsü Şekil 5.6'da verilmiştir.

Çalışma sırasında grafik çizimleri için geribeslemeli kontrol sisteminden alınması gereken veriler Quartus II içinde bütünleşik olarak bulunan SignalTapII Logic Analyzer kullanılarak elde edilebilmektedir. Bu aracın ekran görüntüsü de Şekil 5.7'de görülmektedir. SignalTapII Logic Analyzer aracının kullanımı Bölüm 5.7'de detaylı bir şekilde verilmiştir.



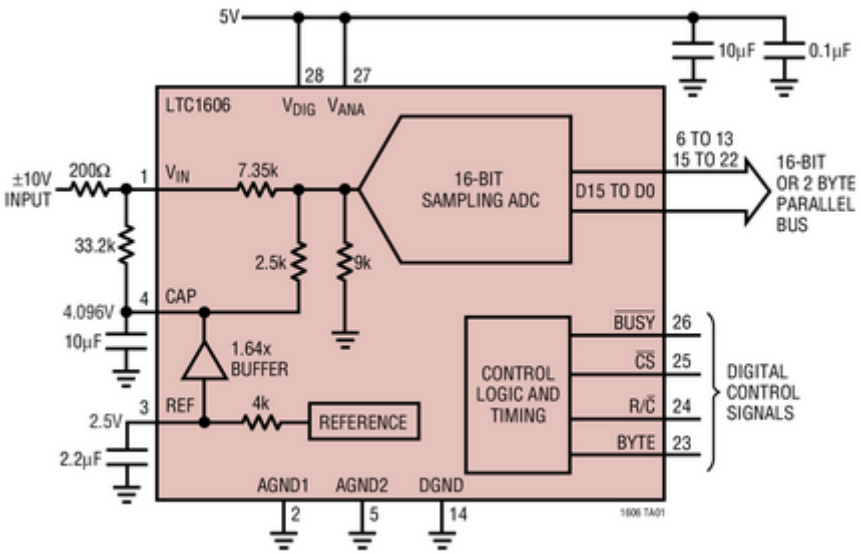
Şekil 5.6: Programmer penceresi ekran görüntüsü



Şekil 5.7: SignalTap II Logic Analyzer penceresi ekran görüntüsü

5.2. Ek Kart Yapısı ve Tasarımı

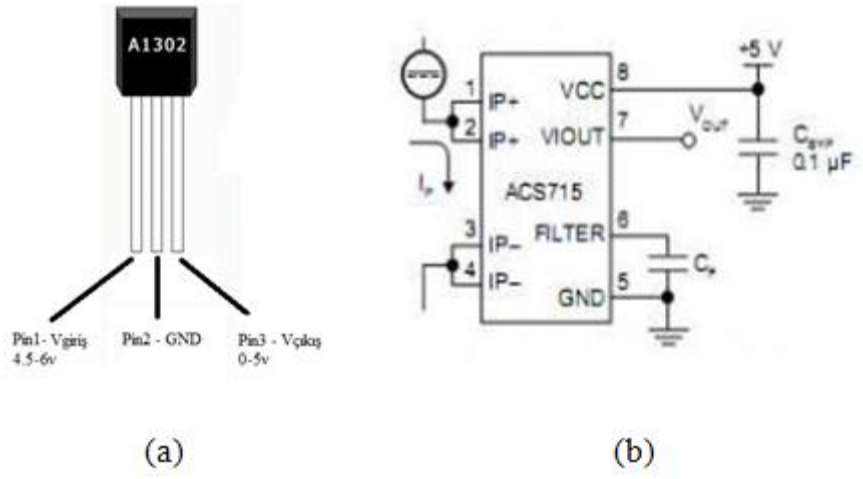
DE2-115 eğitim ve geliştirme setinin 40 iğneli soketine bağlanarak kullanılan ek kart üzerinde iki kanal ADC bulunmaktadır. DC 5 V ile çalıştırılan LTC1606 kodlu ADC tümleşikleriyle yapılmış olan bu ADC kanallarından 250 Ksps (saniyede 250000 örnek) hızında ve 16-bit büyüklüğünde veri elde edilebilmektedir. Bu tümleşik için verilen bilgi sayfasında bulunan ve Şekil 5.8'de görülen, tipik kullanım devresinden faydalanılarak ADC kanalları oluşturulmuştur (Linear Technology 2000). ADC1 kanalı, EMLS bobininin altına yerleştirilmiş olan Hall-effect algılayıcısının çıkış gerilimini sayısal veriye dönüştürmek için kullanılırken, ADC2 kanalı, bobin üzerinden geçen akımı algılayan akım algılayıcısının çıkış gerilimini sayısal veriye dönüştürmek amacıyla kullanılmıştır.



Şekil 5.8: ADC devresi (Linear Technology 2000)

EMLS'de küre mıknatısın bobine olan uzaklığını ölçmek amacıyla kullanılan Hall-effect algılayıcı elemanı Şekil 5.9a'da görülmektedir (Allegro 2015a). DC 5 V kaynak gerilimi ile çalışan bu eleman, üzerine uygulanan manyetik alana bağlı olarak bir çıkış gerilimi üretmektedir. 0-5 V arasında üretilen gerilim uygulanan manyetik alanın yönüne göre 0-2.5 V aralığında veya 2.5-5 V aralığında olabilmektedir. Bu tez çalışmasında, Hall-effect algılayıcı, elektromıknatıs olarak kullanılan bobinin alt ortasına ve ön yüzü dışa gelecek

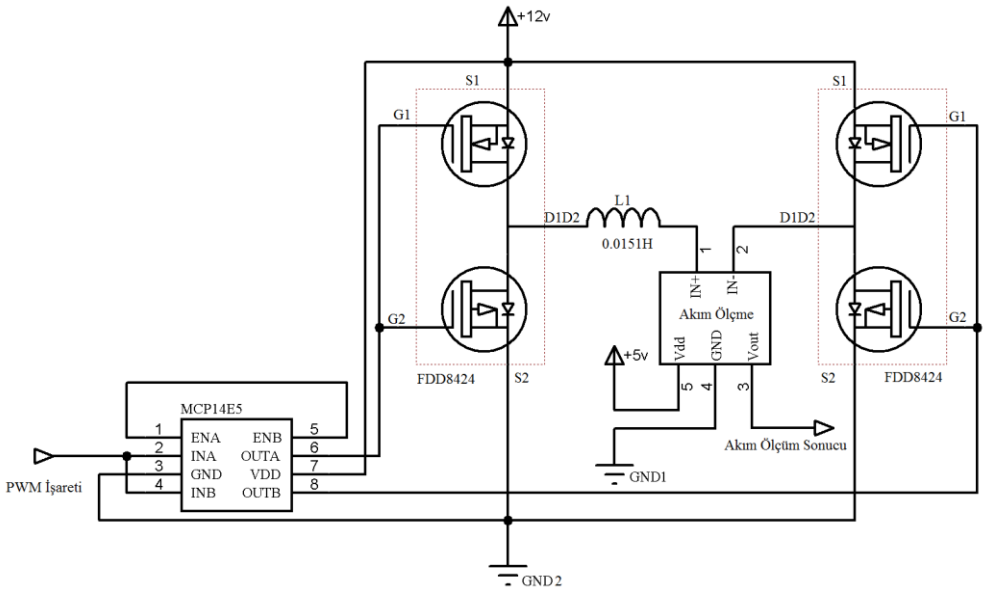
şekilde yerleştirilmiş ve böylece uygulanan manyetik alanın 2.5-5 V aralığında çıkış gerilimi oluşturması sağlanmıştır. Küre mıknatısın bobine yaklaşması algılayıcının çıkış geriliminin artmasına, uzaklaşması ise azalmasına yol açmaktadır. Algılayıcı, küre mıknatısın hareketinden etkilenirken aynı zamanda bobinin oluşturduğu manyetik alandan da etkilenmektedir. Bobin üzerindeki manyetik alan değişiminin algılayıcı çıkış gerilimine etkisi (5.2) denkleminde görülmektedir. Buna göre algılayıcının çıkış gerilimi küre mıknatıs ile bobin arasındaki mesafenin karesi ile ters orantılı ve bobinden geçen akım ile doğru orantılı olarak değişmektedir (Zeltom 2015).



Şekil 5.9: a) Hall-effect algılayıcı(Allegro 20015a), b) Akım algılayıcı (Allegro 2015b)

Ek kartın üzerinde bulunan diğer bir bölüm de, 4 adet MOSFET ve bir MOSFET sürücüsünden oluşan H-bridge devresidir. Bu devre, her örnekleme adımında RKMPC/RKPE algoritması tarafından bulunan yeni kontrol işaretini sisteme uygulayabilmek için kullanılmıştır. Şekil 5.10'da verilmiş olan devre şemasından görüldüğü gibi, bir MOSFET sürücüsü MPC14E5 (Michrochip 2008) ve iki adet çift MOSFET'li FDD8424 (Fairchild Semiconductor 2015) elemanlarından oluşan bobin sürücü devresinin kaynak gerilimi (+12 V), ADC kanalları ve DE2-115'in kaynak geriliminden (+5 V) ayrılmıştır. Böylece ADC kanalları ve DE2-115, bobin ve bobin sürücü devresinde oluşabilecek bozucu etkilerden korunmuştur. Bobin üzerinden geçen akımı ölçmek için kullanılan ve bobine seri olarak bağlanmış olan akım algılama devresinin şeması da Şekil 5.9b'de görülmektedir (Allegro 2015b). Akım ölçme devresi, 2.5 V değeri 0 A'i ifade etmek üzere, (-) yönde akımlar için 0-2.5 V ve (+) yönlü akımları için de

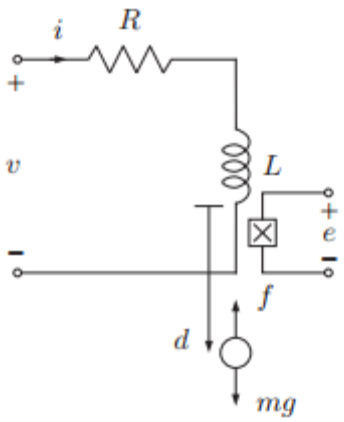
2.5-5 V aralığında bir çıkış gerilimi üreterek, bobin üzerinden geçen akımın iki yönlü olarak ölçülmesine olanak vermektedir.



Şekil 5.10: H-bridge devresi

5.3. Elektromanyetik Askı Sistemi (EMLS)

Şekil 5.11'de sistem modeli görülen elektromanyetik askı sistemi, bir bobin (elektromıknatıs), havada asılı halde tutulacak olan bir küre mıknatıs ve bir Hall-effect algılayıcıdan (sistem çıkışı) oluşmaktadır. Burada R bobin direncini (Ω), L bobin endüktansını (H), i bobin akımını (A), f bobinin küre mıknatısa uyguladığı kuvveti ($^{m}/_{s^2}$), m küre mıknatısın kütlesini (Kg), g yerçekimi katsayısını ($^{m}/_{s^2}$), d bobinin altı ile küre mıknatıs arasındaki uzaklığı (m), ve e Hall-effect algılayıcının ürettiği çıkış gerilimini (V) ifade etmektedir (Zeltom 2015). Bobinin üzerinden geçen akımın yönü ve değeri ile oluşturulan manyetik çekme/itme gücü doğru orantılı olarak değişmektedir. Herhangi bir anda, bobinin, küre mıknatısa uyguladığı kuvvet ile yerçekimi kuvveti eşitlenirse, küre mıknatıs havada hareketsiz bir biçimde (asılı halde) duracaktır.



Şekil 5.11: Elektromanyetik askı sistemi modeli (Zeltom 2015)

Elektromıknatısın küre mıknatısa uyguladığı kuvvet yaklaşık olarak,

$$f = k\frac{i}{d^3} \tag{5.1}$$

şeklinde ifade edilebilir (Cheng 1983). Burada k sistemin geometrik yapısına bağlı olarak oluşan bir parametredir. Sistem çıkışı olan Hall-effect çıkış gerilimi,

$$e = \frac{\beta}{d^2} + \gamma i + \alpha \tag{5.2}$$

şeklinde tanımlanabilir (Smaili ve Mrad 2008). Burada β, γ ve α parametreleri Hall-effect algılayıcıya ve sistemin geometrik özelliklerine bağlıdır ve gürültü ihmal edilmiştir. Yerçekimi ve bobinin, küre mıknatısa uyguladığı kuvvetlere Newton'un ikinci kuralı uygulandığında,

$$m\ddot{d} = mg - k\frac{i}{d^3} \tag{5.3}$$

denklemi ve bobine Kirchhoff'un gerilim kuralı uygulandığında,

$$v = Ri + Li \tag{5.4}$$

denklemi elde edilir. Burada $x = [x_1 \quad x_2 \quad x_3] = [d \quad \dot{d} \quad i]$ ve $y = e$ olarak kabul edilirse EMLS'nin durum uzayı modeli,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ g - \frac{k}{m}\frac{x_3}{x_1^3} \\ -\frac{R}{L}x_3 + \frac{1}{L}u \end{bmatrix} \qquad (5.5)$$

$$y = \frac{\beta}{{x_1}^2} + \gamma x_3 + \alpha$$

şeklinde elde edilir (Zeltom 2015). Sistem çıkışını ifade eden Hall-effect algılayıcının çıkış gerilimi (y), sadece küre mıknatısın yaklaşıp uzaklaşmasından değil, aynı zamanda, kontrol işlemi sırasında bobin üzerinde oluşturulan manyetik alandan ve diğer dış etkenlerden de etkilenmekte olduğu (5.5)'ten açıkça görülmektedir. Bobin üzerine uygulanan gerilimin bobin üzerinde oluşturduğu manyetik alan, bobinden geçen akımın büyüklüğü ile doğru orantılıdır ve bu manyetik alan Hall-effect algılayıcı çıkış gerilimini de aynı yönde etkiler. Yani Hall-effect algılayıcı çıkış gerilimine, bobin üzerinde oluşturulan manyetik alanın etkisi ile küre mıknatısın bobine yaklaşmasının etkisi aynı yönlüdür. Bu durumu şu şekilde açıklamak da mümkündür: Küre mıknatısı bobine yaklaştırmak için bobin üzerindeki manyetik alanı arttırmak gerekir. Bu durumdan etkilenen Hall-effect algılayıcının çıkış gerilimi de artar. Bununla birlikte artan manyetik alan nedeniyle küre mıknatıs bobine daha fazla yaklaşır. Bu da aynı şekilde Hall-effect algılayıcı çıkış geriliminin artmasına yol açar. Bu durum Hall-effect algılayıcı çıkış geriliminin doğrusallığını bozar.

(5.5) ile verilen durum uzayı modelindeki R, L ve m parametrelerinin değerleri, model kullanılmadan önce ölçülerek belirlenebilir fakat, k, β, γ ve α parametreleri ölçülerek belirlenebilecek durumda değillerdir. Dolayısıyla bu parametrelerin değerlerinin belirlenmesi için sırasıyla şu işlemlerin yapılması gerekmektedir (Zeltom 2015):

- Küre mıknatıs bobinden uzaktayken bobin uçlarına 0 V gerilim uygulandıktan sonra, Hall-effect algılayıcı çıkış gerilimi ölçülerek e elde edilir. Buradan $\alpha = e$ kabul edilerek α parametresi elde edilir.

- Küre mıknatıs bobinden uzaktayken bobin uçlarına 1 V gerilim uygulandıktan sonra, Hall-effect algılayıcı çıkış gerilimi ölçülerek e elde edilir. Buradan $e = \alpha + \gamma \frac{v}{R}$ denklemi üzerinden γ parametresi elde edilir.

- Uygun bir kontrolör ile küre mıknatıs bobinden 2 cm uzakta olacak şekilde kontrol edilirken, Hall-effect algılayıcı çıkış gerilimi ölçülerek e elde edilir. Buradan $e = \frac{\beta}{d^2} + \gamma i + \alpha$ denklemi üzerinden β parametresi elde edilir.

- Daha sonra $mg = k \frac{v/R}{d^3}$ üzerinden de k parametresi bulunur.

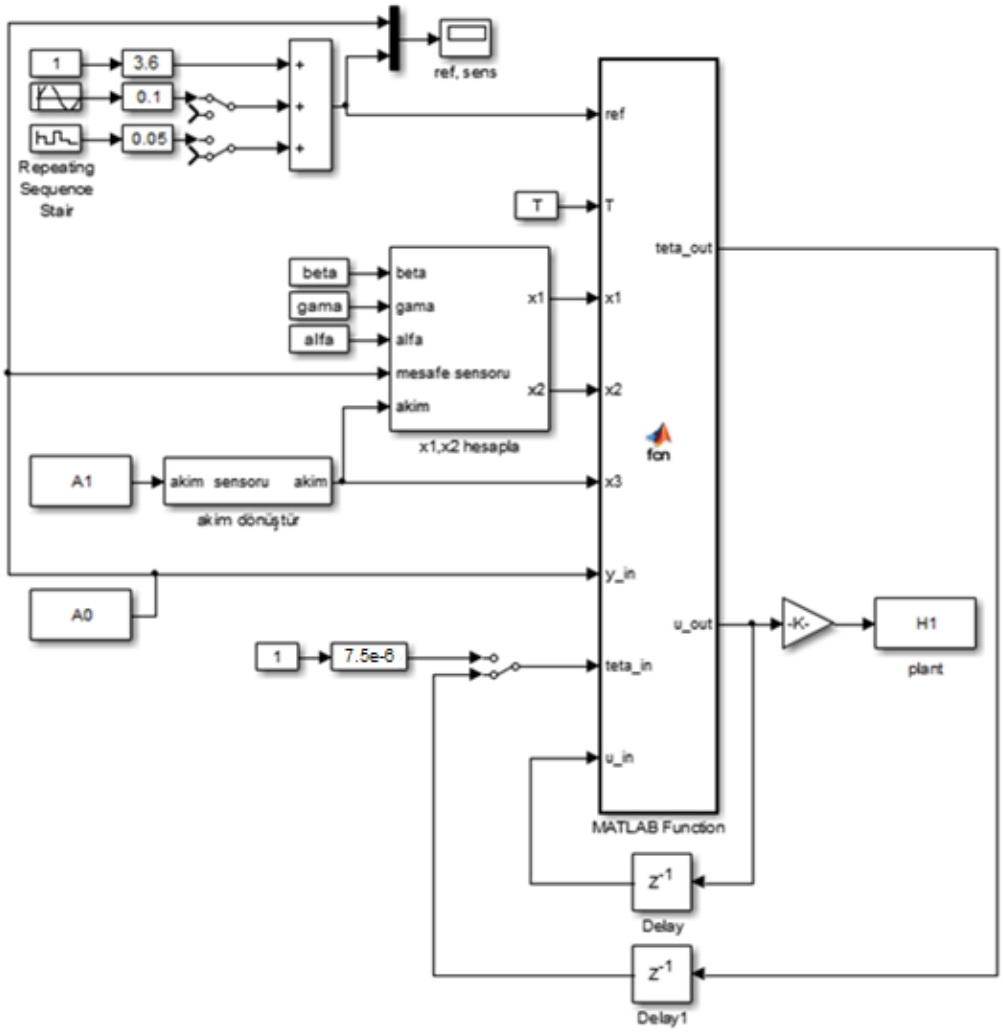
Bu işlemlerin yapılması sonucunda tez çalışması sırasında sistem parametrelerinin kullanılan değerleri Tablo 5.1'de verilmiştir.

Tablo 5.1: EMLS parametreleri

Parametre	**Değer**
R	1.7 [Ω]
L	0.0151 [H]
g	9.81 [m/s^2]
m	0.0413 [kg]
k	3.1×10^{-6} [A^2×kg×m^5/s^2]
β	4.25×10^{-4} [V/m^2]
γ	0.31 [V/A]
α	2.48 [V]

5.4. RKMPC/RKPE Algoritmasının MATLAB/Simulink ile Gerçeklenmesi

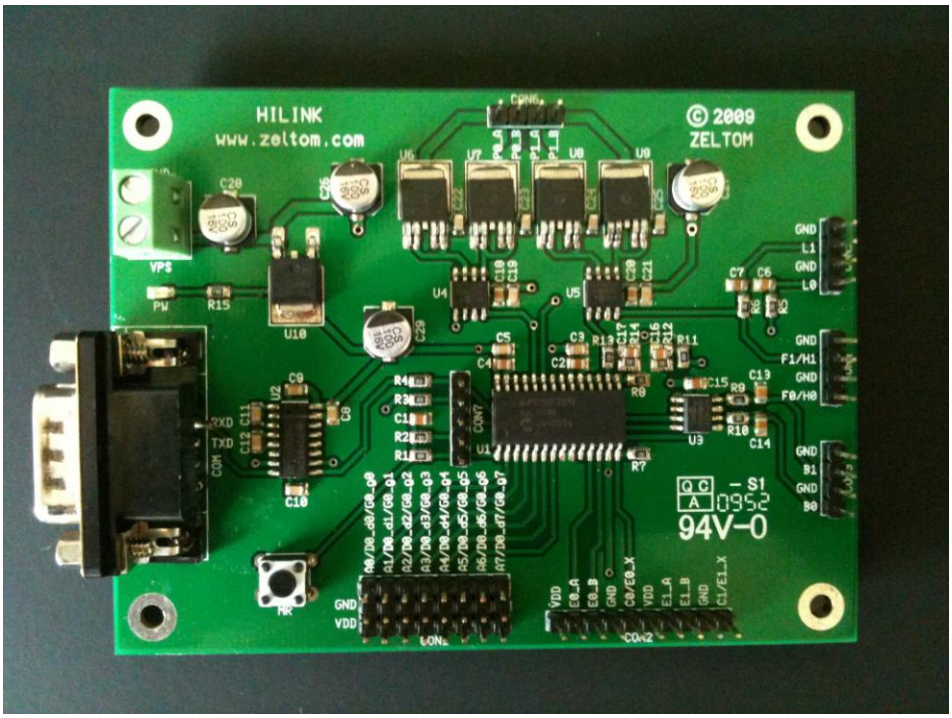
FPGA üzerinde gerçeklenmesinden önce RKMPC ve RKPE algoritmalarının MATLAB üzerinde benzetimleri yapılmıştır. Daha sonra Simulink üzerinden EMLS'nin gerçek-zamanlı kontrolü gerçekleştirilmiştir. Simulink ile oluşturulan modelin görüntüsü Şekil 5.12'de görülmektedir. Algoritma kodları kullanıcı tanımlı fonksiyon (User Defined Function - Fcn) aracı içerisine yazılmıştır.



Şekil 5.12: MATLAB/Simulink modeli

EMLS çıkışı olan Hall-effect algılayıcının ürettiği gerilimi ve bobin üzerinden geçen akımı ölçmek için kullanılan akım ölçme devresinin ürettiği gerilimi ölçmek ve bunun yanı sıra RKMPC/RKPE algoritmasının ürettiği kontrol işaretini bobine aktarmak için Şekil 5.13'te görülen Hilink kartı kullanılmıştır. Şekil 5.12'de görülen MATLAB/Simulink dosyasında bulunan A1 kutusu, Hilink kartı üzerinde, EMLS çıkış geriliminin elde edildiği ADC kanalının, A2 kutusu da bobin akımının elde edildiği ADC kanalının bilgisini MATLAB/Simulink dosyasına taşımaktadır. H1 kutusu ise, Hilink kartı üzerinde bulunan ve üretilen kontrol işaretini EMLS'ye aktaracak bobin sürücü devresine bilgi aktarmayı sağlamaktadır (Zeltom 2015). Kart üzerinde bulunan 8 adet ADC kanalından ikisi algılayıcı çıkış gerilimlerini sayısal bilgiye dönüştürmek için kullanılmıştır.

MATLAB/Simulink ile oluşturulan kontrol işaretini EMLS'ye iletmek için iki adet H-bridge MOSFET devresinden birisi kullanılmıştır.

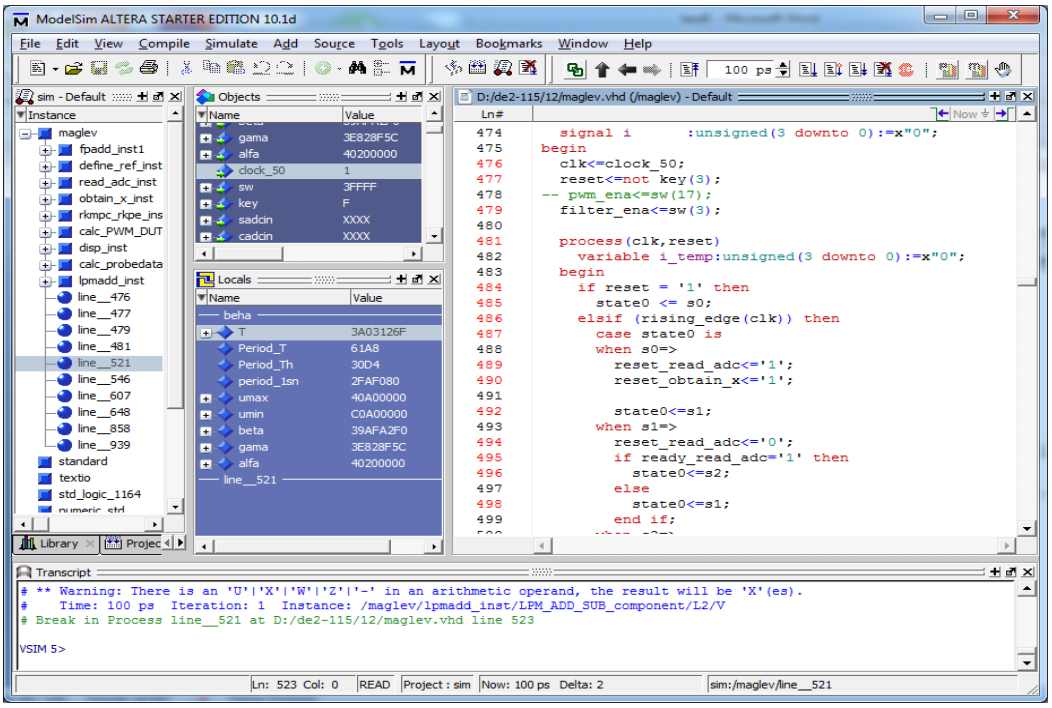


Şekil 5.13: Hilink arabirim kartı (Zeltom 2015)

Bilgisayar ile Hilink kartı arasındaki bağlantı standart 9-pinli seri bağlantı (RS232) üzerinden gerçekleştirilmektedir. Bu nedenle bilgisayar ile Hilink kartı arasındaki iletişim sınırlı bir hızda kalmıştır ve oluşturulan geribeslemeli kontrol sisteminde örnekleme zamanı (T_s) en az 1ms olacak şekilde kullanılabilmiştir. Buna rağmen MATLAB/Simulink modeli, Hilink kartı ve EMLS'den oluşan geribeslemeli kontrol sisteminin başarılı bir şekilde çalıştığı ve EMLS çıkışının, sabit, sinüzoidal ve basamak olmak üzere, uygulanan üç farklı referans işaretini oldukça yakından takip ettiği görülmüştür.

5.5. RKMPC Algoritmasının ModelSim ile Benzetimi

RKMPC algoritmasını FPGA üzerinde oluşturmak için yazılan VHDL kodları öncelikle benzetim programı "ModelSim Altera Starter Edition 10.1d" üzerinde denenmiştir. Bu benzetim programı, yazılan VHDL kodlarını, içerdiği MF'leri de kapsayacak şekilde gerçeği ile birebir benzetebilmekte ve bu sayede yazılımın oluşturulmasında büyük kolaylık sağlamaktadır. Benzetimlerde, her clock darbesi için kodların ve MF'lerin ürettiği sonuçlar değerlendirilebilmekte ve böylece hata tespiti mümkün hale gelmektedir. RKMPC/RKPE algoritmasının içerdiği tüm hesaplamaların sonuçları tek tek denetlenmiş ve doğruluğu MATLAB üzerinde gerçekleştirilen hesaplamalarla karşılaştırılmıştır. Şekil 5.14'te ekran görüntüsü verilen ModelSim Altera benzetim programı üzerinde yapılan bu işlemler sonrasında, RKMPC/RKPE algoritmasının FPGA üzerinde gerçeklemesinin $H = 6$ için yaklaşık olarak 300 µs süre harcadığı tespit edilmiştir. Bu nedenle geribeslemeli kontrol sisteminin örnekleme zamanı $T_s = 0.5\,ms$ olacak şekilde kullanılmıştır. 2.6 GHz çalışma osilatörü frekansına sahip i5-3230M işlemcili bir dizüstü bilgisayar üzerindeki MATLAB ortamında aynı algoritmanın tamamlanma süresi ortalama 3ms'dir.

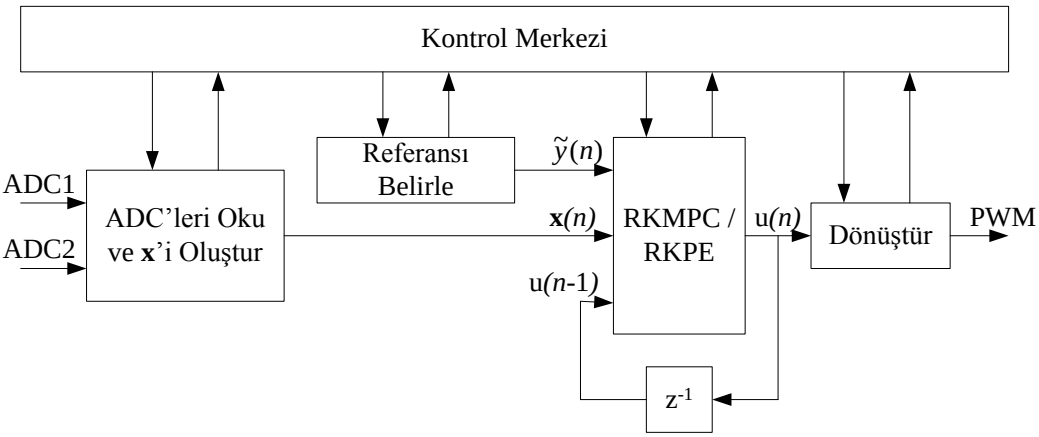


Şekil 5.14: ModelSim altera ekran görüntüsü

5.6. RKMPC Algoritmasının FPGA Üzerinde Gerçeklenmesi

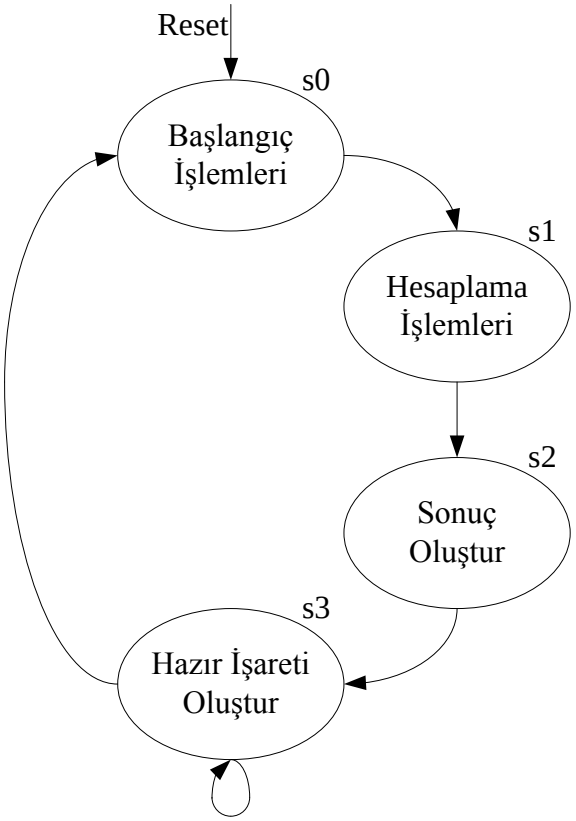
RKMPC algoritmasının FPGA üzerinde gerçeklenmesi için ALTERA firmasının kullanıma sunduğu Quartus II (sürüm 13.0sp1 64-bit) yazılım geliştirme aracı kullanılmıştır. Bu tez çalışmasında VHDL kodlama tekniği kullanılmış ve tüm matematiksel işlemler Quartus II yazılım geliştirme aracı ile beraber sunulan ve 32-bit floating-point (kayan nokta) yapısı kullanan MF'ler kullanılarak gerçekleştirilmiştir. Derleme sonrasında elde edilen FPGA kodlarının DE2-115'ya gönderilebilmesi için, Quartus II içerisinde bütünleşik olarak sunulan Programmer aracı ve DE2-115 üzerinde bütünleşik olarak bulunan USB-Blaster programlama devresi kullanılmıştır. FPGA çalışma osilatörü olarak DE2-115 üzerinde bulunan 50MHz frekanslı kristal osilatör devresi kullanılmıştır. Çalışma sırasında geribeslemeli kontrol sistemine ait gerekli veriler Quartus II içinde bütünleşik olarak bulunan SignalTap II Logic Analyzer kullanılarak elde edilmiştir.

FPGA üzerinde gerçeklenen donanımın genel yapısı Şekil 5.15'de görülmektedir. RKMPC algoritmasının sıralı bir yapıda olması ve FPGA kaynaklarının sınırlı olması nedeniyle donanım, sonlu durum makinesi (Finite State Machine - FSM) yapısı kullanılmış ve birbirleri ile pipelined (ardı ardına, boru hattı gibi bağlanmış) olacak şekilde bağlanmış olan modüllerden oluşturulmuştur. Ana işlem modülleri, 'Referans Belirleme', 'ADC'leri Oku ve **x**'i Oluştur, 'RKMPC/RKPE ' ve 'Dönüştür' şeklinde adlandırılmıştır ve tüm bu modüller 'Kontrol Merkezi' olarak adlandırılan asıl modül tarafından yönetilmiştir. En alt seviyeli temel modüller ise, MF'lerin FSM yapısı içinde diğer modüllerle pipelined bir yapı olarak kullanılabilmesini sağlayacak şekilde oluşturulmuştur.



Şekil 5.15: RKMPC/RKPE donanımı genel yapısı

Tüm modüller Şekil 5.16'da görüldüğü gibi, 4-adımlı bir FSM yapısı içerecek şekilde oluşturulmuştur. Buna göre birinci adımda 'Reset' tetiklemesi ile başlangıç işlemleri yapılmakta ve MF kendi özgün işlemini yapmaya hazır hale getirilmektedir. İkinci adımda, gerekli hesaplama işlemleri gerçekleştirilmekte ve bu adımın toplam süresi MF'nin kendi özgün işlemini yapmak için ihtiyaç duyduğu saat tetiklemesine (clock pulse) bağlı olarak değişmektedir. Yani bir MF'nin çalışmaya başlaması ile sonucu üretmesi arasında belirli bir zaman gecikmesi (latency) oluşmaktadır. MF'lerin latency miktarı da seçimlik bir bilgidir fakat latency miktarındaki değişmeler, MF'lerin kaynak kullanımını da etkilemektedir.



Şekil 5.16: Modüllerin FSM yapısı

Bu tez çalışmasında kullanılan MF'ler için seçilen latency miktarları ve bunun sonucunda oluşan kaynak kullanım bilgileri Tablo 5.2'de verilmiştir. Bu seçimler yapılırken, hem işlemlerin olabildiğince hızlı gerçekleştirilmesi, hem de daha az kaynak kullanımının sağlanması hedeflenmiştir. Tablo 5.2'de görülen *lut*, sabit veri çizelgesini (look-up table), *reg*, bütünleşik yazmaçları (registers), *lpm_add_sub*, fixed-point toplama çıkarma elemanını, *lpm_mult*, fixed-point çarpma elemanını, *lpm_compare*, fixed-point karşılaştırma elemanını, *mux21*, 2 giriş 1 çıkışlı çoğullayıcıyı (multiplexer), *dsp_9bit*, 9-bit uzunluğunda gömülü Digital Signal Processing (DSP) işlemcisini, *M9K*, gömülü dahili belleği ifade etmektedir.

Tablo 5.2: MF latency ve kaynak kullanım miktarları

MF	**Latency**	**Kaynak Kullanımı**
Toplama (Altfp_add_sub)	7	170 *lut* + 377 *reg*
Çarpma (Altfp_mult)	5	4 *lpm_add_sub* + 1 *lpm_mult* + 136 *reg*
Bölme (Altp_div)	14	16 *dsp_9bit* + 194 *lut* + 1 *M9K* + 74 *mux21* + 970 *reg*
Karekök (Altfp_sqrt)	28	370 *lut* + 1433 *reg*
Sinüs (Altfp_sin)	36	32 *dsp_9bit* + 6536 *lut* + 3554 *reg*
Karşılaştırma (Altfp_copmare)	3	113 *lut* + 19 *reg*
Dönüştürme (Altfp_convert)	8	5 *lpm_add_sub* + 4 *lpm_compare* + 281 *reg*

FSM'nin üçüncü adımında, modülün çıkış bilgisi, bir sonraki modülün kullanabileceği şekilde hazır hale getirilir. Dördüncü ve son adımda ise üçüncü adımda, hazırlanan çıkış verisinin, bir sonraki modül tarafından güvenli bir şekilde kullanılabilmesi için hazır olduğunu belirten 'Ready' işareti üretilir. Her modülde, işlem yoğunluğuna göre belirli bir latency oluşmaktadır. Şekil 5.15'de verilen genel donanım yapısı içindeki ana elemanların latency miktarları ve kaynak kullanım bilgileri Tablo 5.3'te görülmektedir.

Tablo 5.3: RKMPC/RKPE genel donanım birimlerinin kaynak kullanım miktarları

Blok Adı	**Latency**	**Kaynak Kullanımı**		
		LEs	**MBs**	**EMs**
Referans Belirleme	19 (Sabit Ref.)	9148	1390	38
	121 (Basamak Ref.)			
	170 (sinüs Ref.)			
ADC oku ve **x**'i Belirle	1143	3415	2137	23
RKMPC	9657	50954	21313	351
Dönüştür	238	1924	5029	23

Her örnekleme adımında önce 'ADC oku ve **x**'i belirle' bloğu bobinin altına yerleştirilmiş olan Hall-effect algılayıcının çıkış gerilimini ve akım ölçme

devresinin çıkış gerilimini sayısal veriye dönüştürür. Bunun için ek kart üzerinde bulunan LTC1606 ADC tümleşikleri kullanılmıştır. Bu tümleşiğin Chip Select (CS) bacağı GND hattına bağlanarak tümleşiğin sürekli çalışır durumda olması sağlanmıştır. Aynı zamanda BYTE bacağı da GND hattına bağlanarak üretilen sayısal verinin 16-bit büyüklüğünde olması sağlanmıştır. Ek kart üzerindeki her iki ADC kanalı için aynı şekilde yapılmış olan bu biçimlendirmeler altında 'ADC oku ve **x**'i belirle' bloğunda Read/Convert ($R/\overline{C}$) bacakları varsayılan değer olarak mantıksal 1 seviyesinde olacak şekilde tutulmaktadır. 40ns boyunca bu bacakların mantıksal 0 seviyesinde tutulması ile LTC1606 içinde analog gerilimin sayısal bilgiye dönüştürülmesi işlemi başlatılır. Bu süre sonunda $R/\overline{C}$ bacakları tekrar varsayılan hali olan mantıksal 1 seviyesine getirilir. Böylece LTC1606 tümleşikleri sayısal bilgiyi hazırlamaya başlar ve yaklaşık 8 µsn sonra veri çıkış bacaklarından ADC1 ve ADC2 kanallarına ait sayısal veriler elde edilmiş olur. FPGA tarafından elde edilen bu ham sayısal bilgilerin hesaplamalarda kullanılabilmesi için gerçek gerilim değerleri haline getirilmesi gerekir. LTC1606'nın çıkışındaki 16-bit büyüklüğündeki sayısal veri ±10 V'luk bir aralığı ifade etmektedir. Bu durumda en soldaki bitin işaret biti olduğu düşünülürse, sayısal ifadenin 15-bit büyüklüğünde olduğu görülür. Bu ise 0.000305176 V'luk bir hassasiyete karşılık gelmektedir (Linear Technology 2000). Bu nedenle LTC1606 çıkışından elde edilen ham sayısal veri, gerçek gerilim değerine dönüşümün yapılması için, 0.000305176 çarpanı ile çarpılmıştır. Sistem çıkışını ifade eden Hall-effect algılayıcının çıkış gerilimi bu şekilde elde edildikten sonra kullanılabilir fakat akım ölçme devresinin çıkışından benzer şekilde elde edilen çıkış gerilimi değerinin, bobin akımını ifade edecek şekilde dönüştürülmesi gerekir (Allegro 2015b). Bu dönüşüm (5.6) ile verilmiştir.

$$\text{Bobin akımı (A)} = 5 * (\text{Ölçülen gerilim} - 2.5) \qquad (5.6)$$

Her iki ADC kanalından gelen bilgilerin oldukça gürültülü olması nedeniyle 'ADC oku ve **x**'i belirle' bloğuna isteğe bağlı olarak kullanıma sokulabilen sayısal filtreleme işlemi eklenmiştir. Bu filtreleme işlemi, ayarlanabilen belirli bir adet ölçüm sonucunun ortalaması alınarak gerçekleştirilmektedir. Filtreleme işlemi ile ilgili daha detaylı bilgi Bölüm 6'da verilmiştir.

ADC okuma bölümünden elde edilen bu veriler ile sistem denklemlerinde bulunan $x_3[n]$ ve $y[n]$ değerleri elde edilmiştir. Bu veriler (5.5) ile verilen çıkış denkleminde yerine koyularak $x_1[n]$ değeri,

$$x_1[n] = \sqrt{\frac{\beta}{y[n] - \gamma x_3[n] - \alpha}} \qquad (5.7)$$

şeklinde elde edilmiştir. Sistem durum denklemlerinden $\dot{x}_1$ denkleminin Euler yaklaşıklığı kullanılarak çözümü sayesinde $x_2[n]$ de şu şekilde elde edilmiştir:

$$x_2[n] = \frac{x_1[n] - x_1[n-1]}{T_s} \qquad (5.8)$$

'Referans Belirle' bloğunda, içinde bulunulan örnekleme adımı için geçerli olan kestirim ufku boyunca referans işaretinin alacağı değerler belirlenmektedir. Bu sayede RKMPC/RKPE algoritmasının ihtiyaç duyduğu **x** vektörü ve $\tilde{y}$ vektörü elde edilmektedir ve bir önceki örnekleme adımındaki kontrol işareti $(u[n-1])$ ile birlikte bu bilgiler RKMPC/RKPE bloğuna giriş olarak verilmektedir.

RKMPC/RKPE bloğunda uygulanan algoritma Şekil 5.17'te görülmektedir. RKMPC/RKPE algoritması, sıralı yapısı nedeniyle zaman almakta ve yoğun hesaplama yükü nedeniyle de fazla kaynak tüketmektedir. Bu nedenle, FPGA gerçeklemesi yapılmadan önce algoritma, en az sürede tamamlanacak ve en az kaynak kullanımını sağlayacak şekilde optimize edilmiştir. Bu nedenle kimi zaman bir bloğun tekrarlamalı bir yapı içinde kullanımı ile kaynak tüketiminde azalma sağlanmış, kimi zaman da birbirinden bağımsız işlemler aynı zaman diliminde ele alınarak algoritmanın tamamlanma süresi kısaltılmıştır.

1. $u[n] \leftarrow u[n-1]; \hat{\mathbf{x}} \leftarrow \mathbf{x}[n]; \theta \leftarrow \theta[n]$
2. **for** *h*=1:*H*
3. **for** *i*=1:4
4. $\mathbf{f}_i$ hesapla (5.5)
5. $\mathbf{k}_i$ hesapla (4.19)
6. $\frac{\partial \mathbf{k}_i}{\partial x}, \frac{\partial \mathbf{k}_i}{\partial u}, \frac{\partial \mathbf{k}_i}{\partial \theta}$ hesapla (4.21, 4.24, 4.37)
7. **end**
8. $\frac{\partial \hat{\mathbf{f}}}{\partial x}, \frac{\partial \hat{\mathbf{f}}}{\partial u}, \frac{\partial \hat{\mathbf{f}}}{\partial \theta}$ hesapla (4.18, 4.22, 4.36)
9. $\frac{\partial \hat{\mathbf{x}}}{\partial u}, \frac{\partial \hat{\mathbf{x}}}{\partial \theta}$ hesapla (4.17, 4.33)
10. $\hat{\mathbf{x}}[n+h]$ hesapla (4.11)
11. $\hat{y}[n+h]$ hesapla (5.5)
12. $\frac{\partial \hat{y}[n+h]}{\partial u}$ hesapla (4.16)
13. **end**
14. $\delta u[n], \frac{\mathbf{J}_\theta^T \mathbf{e}}{\mathbf{J}_\theta^T \mathbf{J}}$ hesapla (4.10, 4.30)
15. $u^*[n] \leftarrow u[n] + \mu \delta u[n]; \theta[n+1] = \theta[n] - \frac{\mathbf{J}_\theta^T \mathbf{e}}{\mathbf{J}_\theta^T \mathbf{J}_\theta}$ (4.26, 4.30)

Şekil 5.17: RKMPC/RKPE algoritması

Kestirim ufku boyunca her adımda öncelikle, 4.-dereceden RK yöntemi ile sistemin RK modeli kullanılarak bir sonraki konumuna ilerletilmesi gerekmektedir. Bu amaçla (4.19) ile verilen RK parametrelerinin (**k**) bulunması gerekir. (4.19) incelendiğinde bu parametrelerin bir önceki parametreye bağlı olarak hesaplanması gerektiği görülür. Ayrıca (4.21, 4.24, 4.37) denklemleri ile verilen bu parametrelere ait türev işlemleri de benzer şekilde bir öncekine bağlı olarak hesaplanmaktadır. Dolayısıyla bu işlemler bir döngü içinde tekrarlanan bir yapı olarak ele alınabilir. Bunun için (4.19, 4.21, 4.24, 4.37) denklemleri sırasıyla,

$$\begin{aligned} \mathbf{k}_i &= T_s \mathbf{f}(\hat{\mathbf{x}}[n+h] + a \times \mathbf{k}_{i-1}, u) \\ \frac{\partial \mathbf{k}_i}{\partial \mathbf{x}} &= T_s \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+a\times\mathbf{k}_{i-1}} \left(\mathbf{I} + b \frac{\partial \mathbf{k}_{i-1}}{\partial \mathbf{x}} \right) \right) \end{aligned} \qquad (5.9)$$

$$\frac{\partial \mathbf{k}_i}{\partial u} = T_s\left(b\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+a\times\mathbf{k}_{i-1}} \frac{\partial \mathbf{k}_{i-1}}{\partial u} + \frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+a\times\mathbf{k}_{i-1}}\right)$$

$$\frac{\partial \mathbf{k}_i}{\partial \theta} = T_s\left(\frac{\partial \mathbf{f}}{\partial \theta} + b\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_{i-1}}{\partial \theta}\right)\Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h]+a\times\mathbf{k}_{i-1} \\ u=u[u]}}$$

şeklinde düzenlenir ve her adımda hesaplama başlamadan önce a ve b parametreleri uygun şekilde ayarlanır. Bunun için $i=1$ için $a=0$ ve $b=0$, $i=2$ için $a=0.5$ ve $b=0.5$, $i=3$ için $a=0.5$ ve $b=0.5$ ve son olarak $i=4$ için $a=1$ ve $b=0.5$ olacak şekilde belirlenir. Bu durumda i=1 için,

$$\begin{aligned}
\mathbf{k}_1 &= T_s\mathbf{f}(\hat{\mathbf{x}}[n+h]+0\times\mathbf{k}_0, u) \\
&= T_s\mathbf{f}(\hat{\mathbf{x}}[n+h], u) \\
\frac{\partial \mathbf{k}_1}{\partial \mathbf{x}} &= T_s\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0\times\mathbf{k}_0}\left(\mathbf{I}+0\frac{\partial \mathbf{k}_0}{\partial \mathbf{x}}\right)\right) \\
&= T_s\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]}\right) \\
\frac{\partial \mathbf{k}_1}{\partial u} &= T_s\left(0\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0\times\mathbf{k}_0}\frac{\partial \mathbf{k}_0}{\partial u} + \frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0\times\mathbf{k}_0}\right) \\
&= T_s\left(\frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]}\right) \\
\frac{\partial \mathbf{k}_1}{\partial \theta} &= T_s\left(\frac{\partial \mathbf{f}}{\partial \theta} + 0\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_0}{\partial \theta}\right)\Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0\times\mathbf{k}_0 \\ u=u[u]}} \\
&= T_s\left(\frac{\partial \mathbf{f}}{\partial \theta}\right)\Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h] \\ u=u[u]}}
\end{aligned} \qquad (5.10)$$

ve i=2 için,

$$\begin{aligned}
\mathbf{k}_2 &= T_s\mathbf{f}(\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_1, u) \\
\frac{\partial \mathbf{k}_2}{\partial \mathbf{x}} &= T_s\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_1}\left(\mathbf{I}+0.5\frac{\partial \mathbf{k}_1}{\partial \mathbf{x}}\right)\right) \\
\frac{\partial \mathbf{k}_2}{\partial u} &= T_s\left(0.5\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_1}\frac{\partial \mathbf{k}_1}{\partial u} + \frac{\partial \mathbf{f}}{\partial u}\Big|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_1}\right) \\
\frac{\partial \mathbf{k}_2}{\partial \theta} &= T_s\left(\frac{\partial \mathbf{f}}{\partial \theta} + 0.5\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_1}{\partial \theta}\right)\Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_1 \\ u=u[u]}}
\end{aligned} \qquad (5.11)$$

ve i=3 için,

$$\mathbf{k}_3 = T_s\mathbf{f}(\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_2, u) \qquad (5.12)$$

$$\frac{\partial \mathbf{k}_3}{\partial \mathbf{x}} = T_s\left(\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_2}\left(\mathbf{I}+0.5\frac{\partial \mathbf{k}_2}{\partial \mathbf{x}}\right)\right)$$
$$\frac{\partial \mathbf{k}_3}{\partial u} = T_s\left(0.5\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_2}\frac{\partial \mathbf{k}_2}{\partial u}+\left.\frac{\partial \mathbf{f}}{\partial u}\right|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_2}\right)$$
$$\frac{\partial \mathbf{k}_3}{\partial \theta} = T_s\left(\frac{\partial \mathbf{f}}{\partial \theta}+0.5\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_2}{\partial \theta}\right)\Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h]+0.5\times\mathbf{k}_2 \\ u=u[u]}}$$

ve i=4 için,

$$\begin{aligned}
\mathbf{k}_4 &= T_s\mathbf{f}(\hat{\mathbf{x}}[n+h]+\mathbf{k}_3, u) \\
\frac{\partial \mathbf{k}_4}{\partial \mathbf{x}} &= T_s\left(\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+\mathbf{k}_3}\left(\mathbf{I}+0.5\frac{\partial \mathbf{k}_3}{\partial \mathbf{x}}\right)\right) \\
\frac{\partial \mathbf{k}_4}{\partial u} &= T_s\left(0.5\left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+\mathbf{k}_3}\frac{\partial \mathbf{k}_3}{\partial u}+\left.\frac{\partial \mathbf{f}}{\partial u}\right|_{\mathbf{x}=\hat{\mathbf{x}}[n+h]+\mathbf{k}_3}\right) \\
\frac{\partial \mathbf{k}_4}{\partial \theta} &= T_s\left(\frac{\partial \mathbf{f}}{\partial \theta}+0.5\frac{\partial \mathbf{f}}{\theta \mathbf{x}}\frac{\partial \mathbf{k}_3}{\partial \theta}\right)\Big|_{\substack{\mathbf{x}=\hat{\mathbf{x}}[n+h]+\mathbf{k}_3 \\ u=u[u]}}
\end{aligned} \qquad (5.13)$$

şeklindedir. Burada $\mathbf{k}_0$=0 dır ve

$$\mathbf{f}(\hat{\mathbf{x}}, u) = \begin{bmatrix} x_2 \\ g-\frac{k}{m}\frac{x_3}{x_1^3} \\ -\frac{R}{L}x_3+\frac{1}{L}u \end{bmatrix} \qquad (5.14)$$

ve

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 3\frac{k}{m}\frac{x_3}{x_1^4} & 0 & -\frac{k}{m}\frac{1}{x_1^3} \\ 0 & 0 & -\frac{R}{L} \end{bmatrix} \qquad (5.15)$$

ve

$$\frac{\partial \mathbf{f}}{\partial u} = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \\ \frac{\partial f_3}{\partial u} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} \qquad (5.16)$$

ve

$$\frac{\partial \mathbf{f}}{\partial \theta} = \begin{bmatrix} \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial \theta} \\ \frac{\partial f_3}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{x_3}{x_1^3} \\ 0 \end{bmatrix} \qquad (5.17)$$

şeklindedir. Böylece RK parametreleri ve bunların türevleri, tek bir modülün 4 kez tekrar çalıştırılması ile elde edilmiş olur ve bu şekilde daha az kaynak kullanılmış olur. Bu işlemler sırasında ve daha sonraki işlemler yapılırken birbirinden bağımsız durumda bulunan tüm işlem bölümleri sıralamadaki yerlerinde aynı anda yani 'paralel işlem' olarak gerçekleştirilir. Şekil 5.17'de verilen algoritmada görüldüğü üzere 6. adımda $\frac{\partial \hat{\mathbf{k}}_i}{\partial x}$, $\frac{\partial \hat{\mathbf{k}}_i}{\partial u}$, $\frac{\partial \hat{\mathbf{k}}_i}{\partial \theta}$, 8. adımda $\frac{\partial \hat{\mathbf{f}}}{\partial x}$, $\frac{\partial \hat{\mathbf{f}}}{\partial u}$, $\frac{\partial \hat{\mathbf{f}}}{\partial \theta}$, 9. adımda $\frac{\partial \hat{\mathbf{x}}}{\partial u}$, $\frac{\partial \hat{\mathbf{x}}}{\partial \theta}$, 14. adımda $\delta u[n]$, $\frac{\mathbf{J}_\theta^T \mathbf{e}}{\mathbf{J}_\theta^T \mathbf{J}_\theta}$ ve 15. adımda $u^*[n] \leftarrow u[n] + \mu \delta u[n], \theta_{n+1} = \theta_n - \frac{\mathbf{J}_\theta^T \mathbf{e}}{\mathbf{J}_\theta^T \mathbf{J}_\theta}$ hesaplamaları paralel olarak gerçekleştirilmiş ve bu sayede algoritmanın tamamlanma süresinde belirgin bir azalma sağlanmıştır.

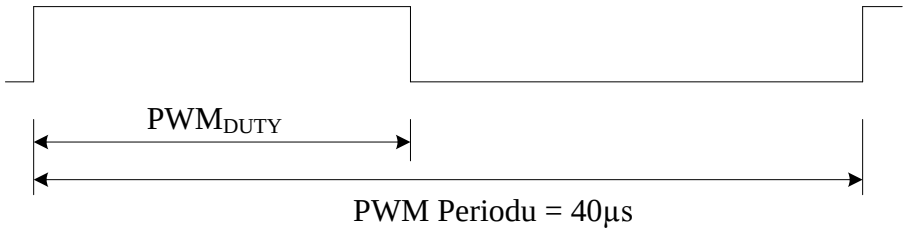
Bölüm 4.2 ve Bölüm 4.3 incelendiğinde RKMPC ve RKPE yapılarının birbiri ile benzer işlemler içerdiği ve aynı süreç içinde değerlendirilebileceği görülür. Bununla birlikte RKPE işleminin kestirim ufku boyunca sadece ilk adımda yapılması ile gerekli değerlerin elde edileceğine ve kestirimin daha sonraki adımlarında değerlendirmeye alınmaması gerektiğine dikkat edilmelidir. Bu nedenle kestirim ufku boyunca ilk kestirim adımında RKMPC ve RKPE işlemleri paralel olarak gerçekleştirilmiş, daha sonra RKMPC işlemlerine devam edilirken RKPE işlemleri ihmal edilmiştir.

Daha önce de belirtildiği gibi, kontrol edilecek sistemin SISO sistem olması nedeniyle (4.10) denkleminde ve kestirimi yapılacak bir parametre için de (4.30) denkleminde bulunan Hessian matrisleri skaler sayı şeklinde oluşmakta ve dolayısıyla matris tersi alma işlemleri basit bölme işlemine dönüşmektedir. Bu durum, algoritmanın FPGA üzerinde gerçekleştirilmesinde kaynak kullanımı açısından büyük bir kolaylık sağlamıştır.

Kestirimi yapılacak parametre olarak önce m parametresi seçilmiş fakat daha sonra, (5.3)'ten de anlaşılacağı üzere k parametresinin değerinin m

parametresine bağlı olarak denge noktası üzerinden elde edilmiş olması nedeniyle, k/m oranının kestiriminin yapılmasına karar verilmiştir. Kontrol işlemi sırasında, hem mevcut küre mıknatısın ağırlığının olduğu gibi kullanılması durumu, hem de küre mıknatısa belirli ağırlıklar eklenmesi durumu için parametrenin değerinin çevrim-içi olarak kestirimi gerçekleştirilmiştir.

Son olarak, 'RKMPC/RKPE' bloğu tarafından oluşturulan kontrol işaretinin ($u[n]$) sisteme uygulanmadan önce PWM işaretine dönüştürülmesi gerekir ve bu işlem 'Dönüştür' bloğunda gerçekleştirilir. Bilindiği gibi, PWM işareti 1 ve 0 değerli iki bölüm içeren, kare dalga şeklindeki bir işarettir ve bu işaretin bir periyodu içinde, 1 değerli bölümün süresinin periyoda oranına PWM_{DUTY} adı verilir. Bununla birlikte, PWM işaretinin periyodunun, sistemin zaman sabitinden çok daha küçük olduğu durumlarda, kontrol işaretinin ($u[n]$) değerinin $[u_{min} - u_{max}]$ aralığının büyüklüğüne oranına eşit bir PWM_{DUTY} oranına sahip bir PWM işaretinin sisteme uygulanması ile, kontrol işaretinin ($u[n]$) sisteme uygulanması sağlanmış olacaktır (Sira-Ramirez 1989; Sira-Ramirez ve Linschinsky-Arenas 1990). Dolayısıyla, kullanılan EMLS'nin zaman sabitinin yaklaşık 9 ms olduğu düşünülerek, uygulanacak PWM işareti, periyodu 40 µs (frekansı 25 KHz) olacak şekilde seçilmiş ve çalışma osilatörü olan 50 MHz frekanslı işaretin 2000'e bölünmesiyle elde edilmiştir. Şekil 5.18'de PWM işaretinin bir periyodu görülmektedir.



Şekil 5.18: PWM işaretinin bir periyodu

Bu bilgiler doğrultusunda, 'Dönüştür' bloğunda önce, kontrol işareti,

$$\text{PWM}_{\text{DUTY}} = \frac{u[n] + u_{max}}{V_s} \times 1000 \qquad (5.18)$$

şeklinde PWM_{DUTY} bilgisine dönüştürülür ve burada $V_s = 12\,V$, $u_{max} = 6\,V$ değerindedir. Böylece kontrol işareti $(u[n])$, 0-1000 değerleri arasında ölçeklendirilmiş olan bir PWM_{DUTY} bilgisine dönüştürülmüştür. Burada PWM_{DUTY} bilgisinin 0 değeri, bobine $u_{min}(-u_{max})$ uygulanmasını, 500 değeri, bobine 0 V uygulanmasını ve 1000 değeri, bobine u_{max} uygulanmasını sağlamaktadır.

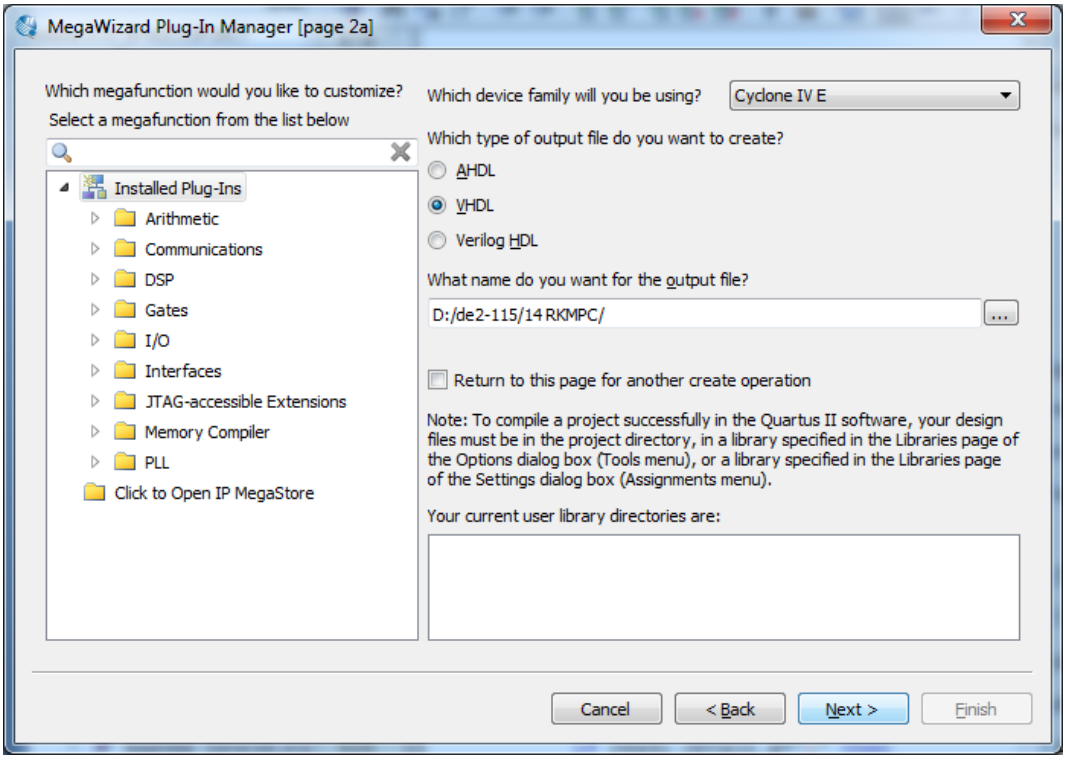
5.7. Geribeslemeli Kontrol Sistemine Ait Verilerin Bilgisayar Tarafında Elde Edilmesi

Geribeslemeli kontrol sisteminin performansının değerlendirilebilmesi amacıyla, her bir örnekleme zamanı için örnekleme zaman indisinin, kontrol edilen sisteme ait durum değişkenlerinin ve çıkış değerlerinin, kontrol işaretinin değerinin, referans işaretinin değerinin ve parametre kestirimi sırasında parametre değerinin bilgisayara gönderilmesi sağlanmıştır. Daha sonra bilgisayar üzerinde çeşitli analizlerin yapılmasına olanak verecek şekilde grafik çizimleri yapılmıştır.

DE2-115 ile bilgisayar arasındaki bağlantı, çeşitli bağlantı yapıları kullanılarak gerçekleştirilebilmektedir. Kart üzerine yerleşik durumda bulunan 2 adet yerel alan ağı (Local Area Network - LAN) devresi, 1 adet standart RS232 bağlantı devresi, 1 adet USB bağlantı devresi bulunmaktadır. Bu devreleri kullanmak için FPGA üzerine buna ilişkin ayrıca bir devrenin kurulması gerekmektedir fakat bu daha fazla kaynak harcanmasına neden olur. Bunun yerine FPGA içerisinde gömülü durumda bulunan In-System Sources and Probes devresi veya SignalTapII Logic Analyzer devresi kullanılabilir. Her iki devre de bilgisayar üzerinden FPGA'nın programlanmasını sağlayan USB bağlantı noktasını kullanan JTAG (Joint Test Action Group) bağlantısını kullanmaktadır.

Bu tez çalışmasında geribeslemeli kontrol sisteminden elde edilen verilerin bilgisayara iletilmesi için öncelikle In-System Sources and Probes devresi kullanılmıştır. In-System Sources and Probes ile izleme yapabilmek için öncelikle, FPGA üzerinde donanımı oluşturmak için kullanılan yazılıma, 1-giriş ve 1-çıkış içeren bir bağlantı noktası eklemek gerekir. Bu işlem, doğrudan kodların yazılması şeklinde yapılabilir, fakat Şekil 5.19'da görülen Mega Wizard

Plug-In Manager penceresi bu işlemin daha kolay bir şekilde gerçekleştirilmesini sağlar. Bu araç ile bağlantı noktasına bir isim verildikten sonra, bağlantının tanıtıcı numarası (ID) tanımlanır ve bit cinsinden gelen-giden bilgi büyüklükleri belirlenir. Bu işlemlerin sonrasında, VHDL yazılımı üzerine, oluşturulan kodların eklenmesi ile işlem tamamlanmış olur. Bu bağlantı kullanılarak, oluşturulan geribeslemeli kontrol sisteminin çalışması sırasında bilgisayar ile karşılıklı veri alışverişi yapılabilmektedir.



Şekil 5.19: MegaWizard Plug-In penceresi

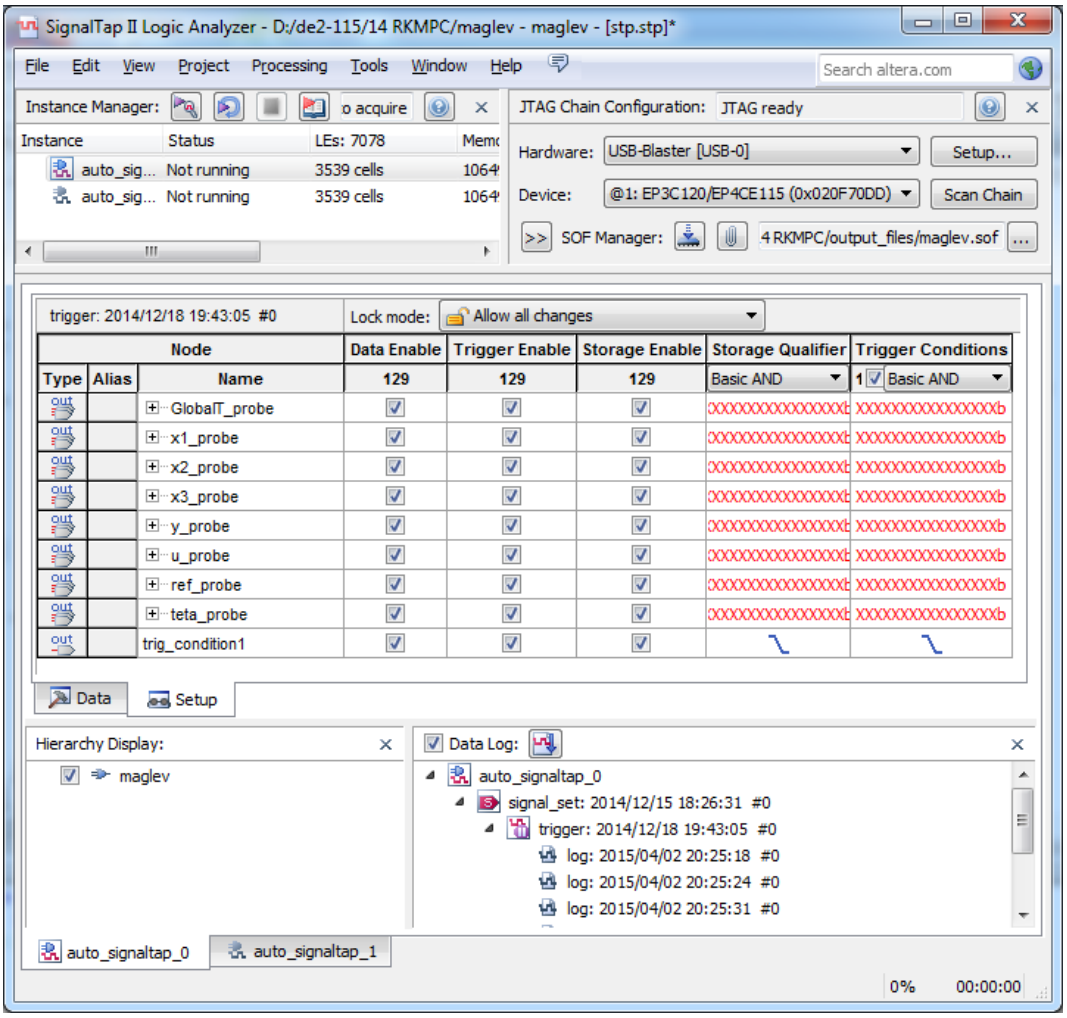
DE2-115 üzerinden bilgisayara gönderilen veriler Quartus II programı üzerinden çalıştırılabilecek olan In-System Sources and Probes Editor penceresi kullanılarak bilgisayarda depolanabilir. Fakat küçük örnekleme zamanlı ve uzun süreli çalışmalar için böyle bir çalışma pek uygun değildir. Bu tür durumlarda, Tools Command Language (TCL) ile kod yazılarak editör penceresi kullanılmadan veri alınabilmektedir. Bu amaçla, Şekil 5.20'de verilen TCL yazılımı, In-System Sources and Probes kullanılarak, DE2-115 üzerinden, tasarlanan geribeslemeli kontrol sistemi ile ilgili verilerin bilgisayara alınması için kullanılmıştır. Fakat burada, örnekleme zamanına bağlı olarak bir veri aktarımı gerçekleştirilememiş ve ancak 4-5ms'lik zaman dilimlerinde veriler bilgisayar

tarafında elde edilebilmiştir. Bu durum bilgisayar tarafında elde edilen her iki veri arasında en az 3 gerçek veri kaybı olduğunu göstermektedir. Bu şekilde elde edilen veriler geribeslemeli kontrol sisteminin davranışını anlamak üzere önemli bilgiler vermiş olmasına rağmen örnekleme zamanına bağlı olarak anlık ve eksiksiz veri temin edilememiştir. Dolayısıyla, gözlemlere dayalı olarak, geribeslemeli kontrol sisteminin başarılı bir performans gösterdiği söylenebilir olsa da, bu durumun grafiksel olarak gösterimi ve ispatı açısından In-System Sources and Probes yöntemi yetersiz kalmıştır.

```
set usb [lindex [get_hardware_names] 0]
set device_name [lindex [get_device_names -hardware_name $usb] 0]

global device_name usb
variable full
start_insystem_source_probe -device_name $device_name -hardware_name $usb

        set output [open myfile.txt w]
        set i 0
        while {$i < 3000 } {
                puts $output [read_probe_data -instance_index 0 -value_in_hex]
                incr i
        }
        close $output

end_insystem_source_probe
```

Şekil 5.20: TCL kodu

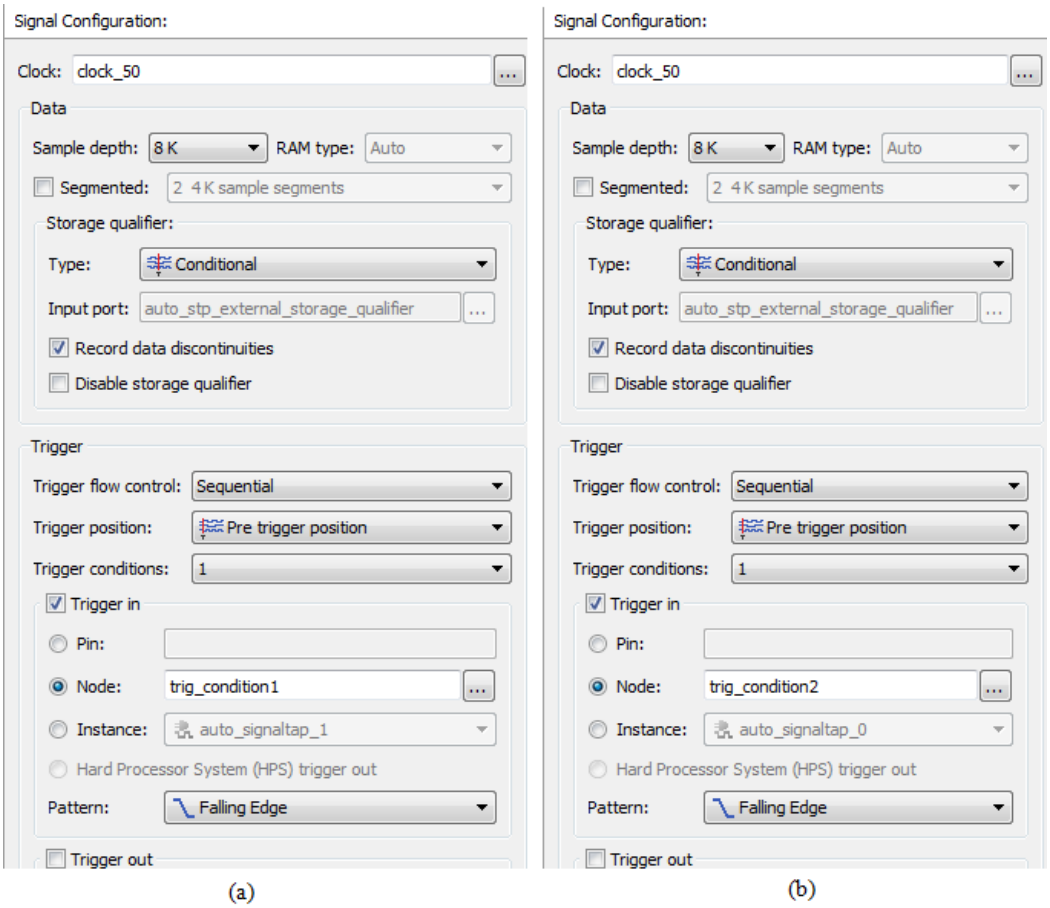
In-System Sources and Probes ile yapılan çalışmanın yetersiz yönleri nedeniyle oluşturulan geribeslemeli kontrol sistemine ait gerekli verilerin bilgisayarda elde edilmesi için SignalTapII Logic Analyzer kullanılmıştır. SignalTapII, FPGA üzerinde gömülü durumda olması nedeniyle daha az kaynak kullanımına ihtiyaç duymaktadır. SignalTapII ile ilgili tüm tanımlamalar Quartus II programı Tools menüsünde SignalTapII Logic Analyzer sekmesi tıklanarak açılan pencerede gerçekleştirilmektedir. Bu pencereye ait görüntü Şekil 5.21'de verilmiştir.



Şekil 5.21: SignalTapII Logic Analyzer penceresi

SignalTapII, elde edilen verileri belirlenen şartlar dahilinde FPGA üzerine gömülü durumda bulunan bellek ortamlarına kaydetmekte ve kayıt alanı dolduğunda bu bilgileri bilgisayara göndermektedir. Bilgisayara gönderilen veriler, burada bir veri toplama alanında sırayla kaydedilmektedir. Bu gönderim belirli bir süreye neden olmakta ve bu süre içinde yine veri kayıpları oluşmaktadır. Bu kayıpları önlemek için öncelikle FPGA üzerindeki bilgilerin bilgisayar aktarılması işlemini yönetmek amacıyla VHDL yazılımında 'trig_condition1' ve 'trig_condition2' adıyla iki veri alanı oluşturulmuştur. Bu alanlar örnekleme zamanına ve FPGA üzerinde SignalTapII'nin bellek kullanımına göre gerekli zamanlarda tetiklemeler vererek, verilerin depolanmasını ve/veya gönderilmesini sağlamıştır. SignalTapII penceresinde Signal Conditions bölümü kullanılarak kayıt belleği 8K büyüklüğünde iki parçalı halde oluşturulmuş ve daha sonra, bu belleklerin kullanımının ardışık şekilde olmasını sağlamak için Şekil 5.22'de

görülen tanımlamalar yapılmıştır. Bu tanımlamalar ve VHDL yazılımına eklenen kodlar ile her örnekleme adımında geribeslemeli kontrol sisteminin çalışması sırasında oluşan veriler aktif durumdaki belleğe sıra ile kaydedilmiş ve kayıt tamamlandığında hem bu bellekteki veriler bilgisayara gönderilmiş, hem de diğer bellek aktif hale getirilerek veri kaybı olmaksızın bu belleğe verilerin kaydı yapılmıştır. Bu işlem dönüşümlü olarak tekrar ettirilmiş ve böylece isteğe bağlı olarak belirlenebilen bir süre için geribeslemeli kontrol sistemine ait gerekli veriler bilgisayara gönderilerek burada depolanmıştır. Ardışık parçalar şeklindeki bu verilerin birleştirilmesi ile elde edilen toplam veri MATLAB üzerine aktarılmış ve tüm analizleri ve grafik çizimleri MATLAB üzerinde gerçekleştirilmiştir.



Şekil 5.22: TrigConditions ve StorageConditions tanımlamaları

6. DENEYSEL SONUÇLAR ve TARTIŞMA

Bu bölümde önce, RKMPC yönteminin FPGA üzerinde gerçeklenmesi ile elde edilen kontrolör kullanılarak, kararsız ve çok hızlı zaman sabitine sahip Doğrusal Olmayan bir sistem olan EMLS'nin kontrolü ile ilgili bilgi verilecek ve elde edilen veriler değerlendirilecektir. Daha sonra, NMPC yöntemi ve RKMPC yöntemi FPGA üzerinde gerçeklenmesi sonucunda elde edilen iki farklı geribeslemeli kontrol sistemine ait veriler üzerinde karşılaştırmalar ve değerlendirmeler yapılacaktır. Son olarak RK tabanlı parametre kestirimi ile gerçekleştirilen Model-Uyarlamalı Öngörülü Kontrol işlemi hakkında bilgiler verilecektir.

6.1. RKMPC Yöntemi ile EMLS Kontrolü

RKMPC yöntemi DE2-115 üzerindeki Cyclone-IV E tip FPGA üzerinde gerçeklenmiş ve üzerinde iki kanal ADC ve bir H-bridge MOSFET devresi ile gerçekleştirilmiş bobin sürücü bulunan ek kart üzerinden gerçek-zamanlı deneysel bir EMLS'nin kontrolü sağlanmıştır. Oluşturulan geribeslemeli kontrol sistemi ile, kontrol edilen sistem çıkışının sabit referans, basamak referans ve sinüzoidal referans işaretlerini takip ettiği gözlemlenmiştir. Geribeslemeli kontrol sistemi üzerinden elde edilen verilerin grafik çizimleri ile de tasarlanan kontrolörün performansı ortaya koyulmuştur.

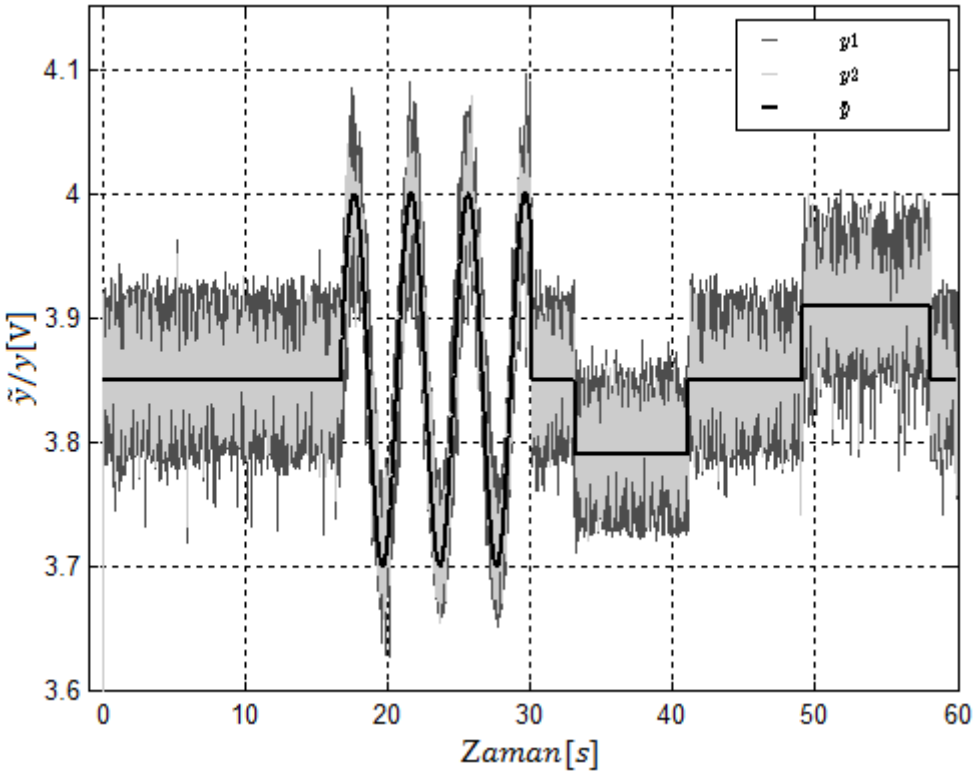
Donanımın gerçeklenmesi sırasında DE2-115 üzerindeki iki konumlu anahtarlara çeşitli görevler tanımlanmıştır. Bu sayede deneysel çalışmalarda DE2-115'in daha etkin olarak kullanımı sağlanmış ve daha kısa sürede daha çok deney yapma olanağı doğmuştur. Nümerik göstergeler anahtarlarla belirlenen parametrelerin görüntülenmesini sağlarken, çıt-çıt anahtarlar çeşitli değer ayarlama işlemlerinde ve kontrolörün RESET edilmesinde kullanılmıştır. DE2-115 üzerinde bulunan bazı elemanlara tanımlanan görevler Tablo 6.1 ile belirtilmiştir.

Tablo 6.1: DE2-115 üzerindeki elemanlara tanımlanan görevler

Eleman Adı	Tanımlanan Görev
SW17	Kontrol işaretinin sisteme uygulanması denetimi
SW16	Nümerik göstergelerde anlık referans değerinin görüntülenmesi/değiştirilmesi
SW15	Nümerik göstergelerde sinüs referans tepe değerinin görüntülenmesi/değiştirilmesi
SW14	Nümerik göstergelerde basamak referans tepe değerinin görüntülenmesi/değiştirilmesi
SW13	Nümerik göstergelerde Hall-effect algılayıcı çıkış geriliminin görüntülenmesi
SW12	Nümerik göstergelerde x_1 değerinin görüntülenmesi
SW11	Nümerik göstergelerde x_2 değerinin görüntülenmesi
SW10	Nümerik göstergelerde x_3 değerinin görüntülenmesi
SW9	Nümerik göstergelerde u değerinin görüntülenmesi
SW8	Nümerik göstergelerde θ değerinin görüntülenmesi
SW7	Nümerik göstergelerde kestirim ufku büyüklüğünün (H) değerinin görüntülenmesi/değiştirilmesi
SW6	Nümerik göstergelerde filtreleme değer adedinin görüntülenmesi/değiştirilmesi
SW5	İşlevsiz
SW4	İşlevsiz
SW3	ADC'lerden okunan bilgilerin filtrelenmesi denetimi
SW2 ... SW0	Eğer 001 ise Sabit referans Değil Eğer 010 ise Sinüs referans Değil Eğer 100 ise Basamak Referans
Disp(7)	İşaret
Disp(6)...Disp(0)	Sayısal Değer (000.0000 biçiminde)
çıt-çıt(3)	Kontrollör Reset
çıt-çıt(2)	İşlevsiz
çıt-çıt(1)	Görüntülenen değeri 1 arttır
çıt-çıt(0)	Görüntülenen değeri 1 azalt

Tasarlanan geribeslemeli kontrol sisteminden gerekli veriler önce In-System Sources and Probes aracı kullanılarak bilgisayar tarafında kaydedilmiş ve

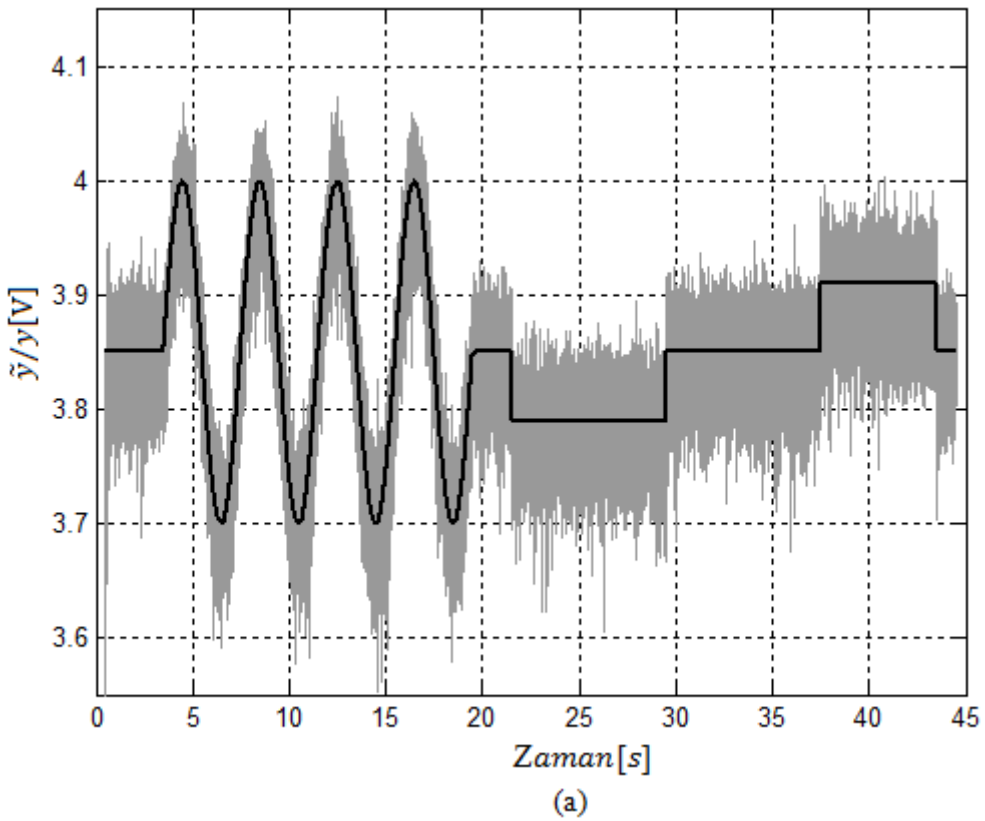
bu verilerin grafik çizimleri MATLAB üzerinde gerçekleştirilmiştir. Daha sonra SignalTapII aracı kullanılarak alınan verilerin MATLAB üzerinde grafik çizimleri gerçekleştirildiğinde, bu verilerin daha gürültülü olduğu gözlemlenmektedir. Bunun sebebi In-System Source and Probe aracı kullanılarak veri elde edilirken veri alma aralığının yaklaşık 4-5 ms olması ve bu durumda oluşan veri kayıplarının, verileri filtrelenmiş gibi bir sonuca götürmesidir. SignalTapII kullanılarak gerçekleştirilen veri toplama işleminde tüm örnekleme adımları için gerekli veriler bilgisayara aktarılmakta ve bu nedenle işaretin filtrelenmemiş doğal hali elde edilmiş olmaktadır. İki veri grubu arasındaki fark Şekil 6.1'de görülmektedir. Burada y1, SignalTapII aracı kullanılarak elde edilen verileri, y2, In-System Sources and Probes aracı kullanılarak elde edilen verileri ve $\tilde{y}$, referans işaretini ifade etmektedir.



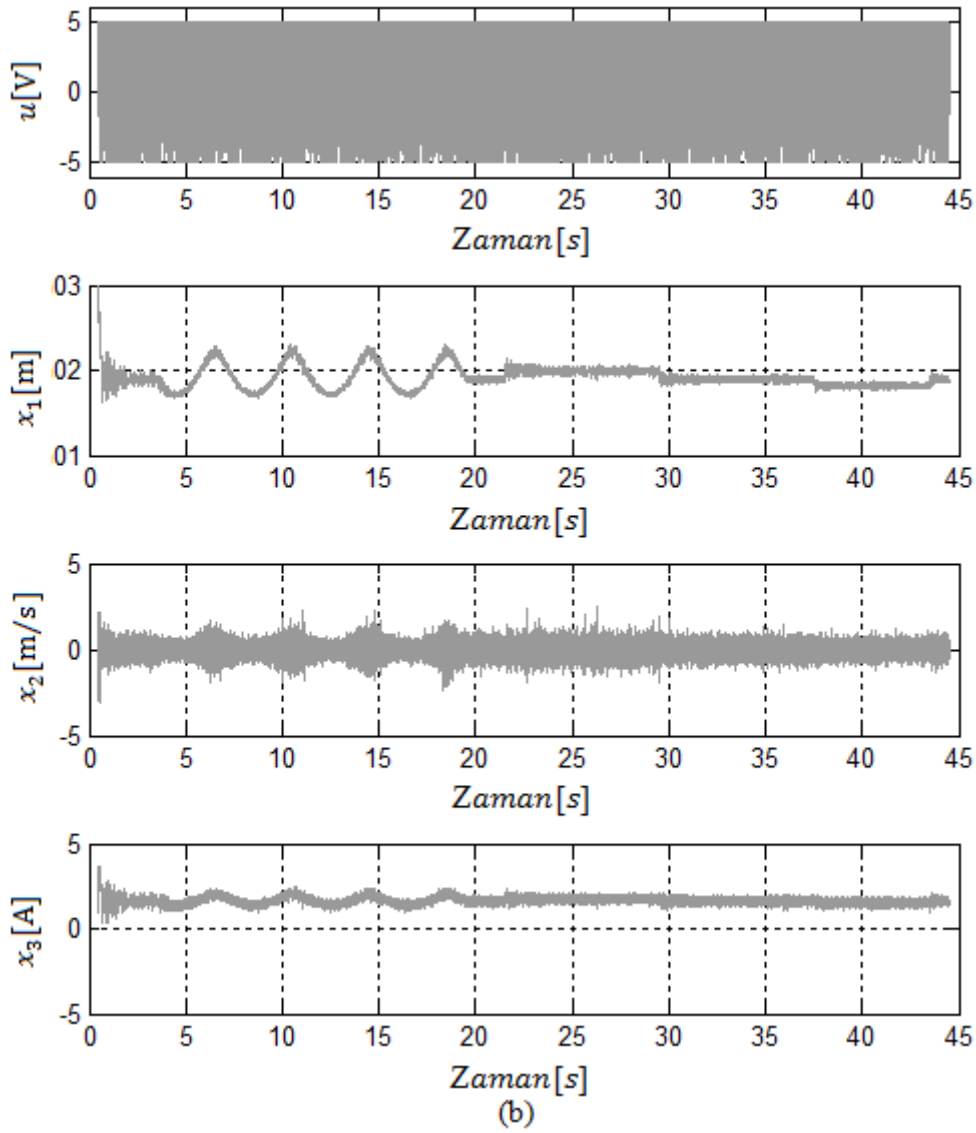
Şekil 6.1: In-System Sources and Probes ve SignalTapII kullanılarak elde edilen verilerin grafik çizimleri

SignalTapII kullanılarak, hem tüm örnekleme adımlarındaki verilerin bilgisayar tarafında elde edilmesi hem de bu verilerin daha az gürültülü hale getirilmesi için 'ADC oku ve **x** belirle' bloğuna bir sayısal filtreleme işlemi eklenmiştir. Bu filtreleme işlemi, '**x** belirle' işleminden önce 'ADC oku' işleminin

isteğe bağlı olarak belirli bir adet tekrar edilmesi ve elde edilen verilerin ortalamalarının '**x** belirle' bloğuna aktarılması esasına dayanmaktadır. Tekrarlama adedinin, DE2-115 üzerinde bulunan iki konumlu anahtarlar ve çıt-çıt anahtarlar kullanılarak farklı değerlere ayarlanması ile elde edilen farklı verilere ait grafik çizimler incelendiğinde, tekrarlama adedi attıkça alınan verilerin daha az gürültülü olduğu görülmüştür. Buna rağmen tekrarlama adedinin artmasının sistem dinamiklerini değiştirdiği ve özellikle mıknatıs topun yerden kaldırılarak doğrusal bölgeye çekildiği ana kadar olan Doğrusal Olmayan kontrolün bundan olumsuz etkilendiği tespit edilmiştir. Ayrıca 10'dan daha fazla miktardaki tekrarlama adetleri için RKPMC/RKPE algoritmasının toplam tamamlanma süresinin, örnekleme zaman aralığını aştığı belirlenmiş ve sonuç olarak filtreleme amacıyla 'ADC oku' işleminin en uygun tekrarlama adedinin 6-7 olduğu saptanmıştır. Şekil 6.2'de filtreleme işlemi yapılmamış ve Şekil 6.3'te de filtreleme işlemi yapılmış duruma ilişkin veriler görülmektedir.



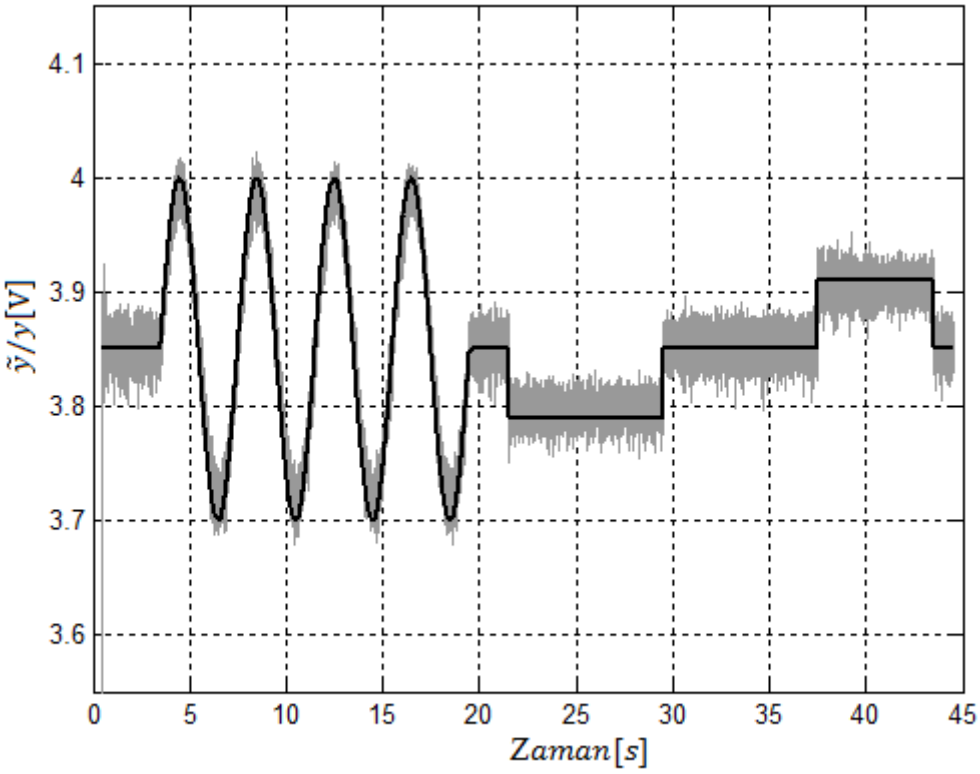
(a)



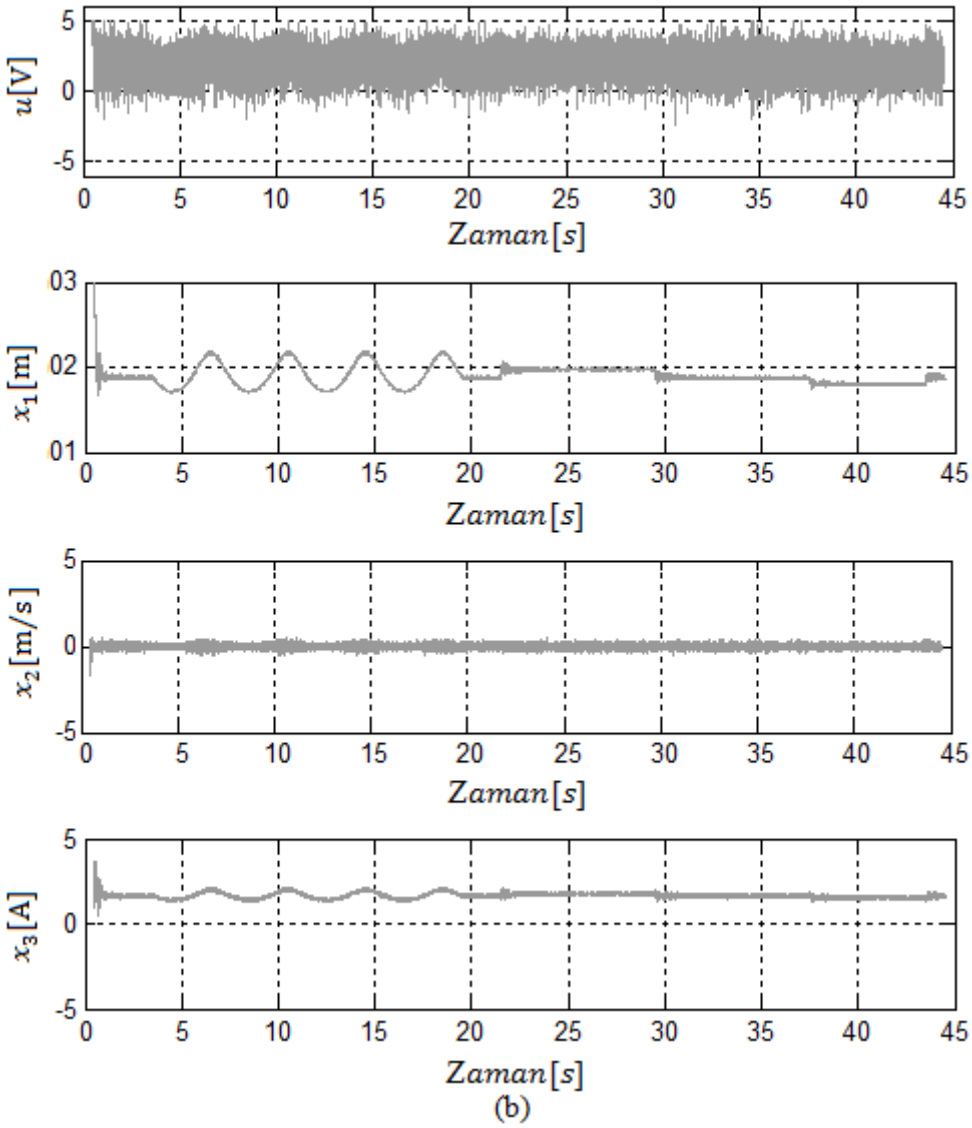
Şekil 6.2: Filtreleme işlemi eklenmemiş geribeslemeli kontrol sistemine ait veriler: (a) Referans ve çıkış işareti, (b) Kontrol işareti ve durum değişkenleri

Şekil 6.2 ve Şekil 6.3 karşılaştırıldığında, öncelikle, filtreleme işlemi eklenmemiş geribeslemeli kontrol sistemine ait grafiklerin tümünde işaretlerin genliklerinin daha büyük yani daha gürültülü olduğu görülmektedir. Şekil 6.2b'de üstten birinci alt grafikte kontrol işaretinin (+) yönde sürekli olarak u_{max} ile sınırlandırıldığı, (-) yönde ise sürekli u_{min}'e yakın noktalarda olduğu görülmektedir. Bu durum filtre eklenmemiş geribeslemeli kontrol sisteminin aşırı yükte çalıştığını ve neredeyse ON/OFF kontrol seviyesinde kaldığını göstermektedir. Oysa ki, Şekil 6.3b'de üstten birinci alt grafikten anlaşılacağı gibi, filtreleme işlemi eklenmiş geribeslemeli kontrol sisteminde kontrol işareti, $u_{min} - u_{max}$ aralığının çok daha dar bir bölümünü kullanmaktadır. Yani filtrelenmiş işaretlerin RKMPC algoritmasında değerlendirilmesi sonucunda daha

uygun kontrol işareti değerlerine ulaşılmaktadır. Bu durumda kontrol işareti üzerinde keskin değişimler yerine daha yumuşak değişimler oluşmaktadır. Küre mıknatısın bobine olan uzaklığındaki değişimi ifade eden x_2 alt grafikleri incelendiğinde Şekil 6.3'te bu alt grafiğin neredeyse 0 noktası etrafında değerler aldığı, buna karşılık Şekil 6.2'de bu alt grafiğin özellikle referans işaretinin ani ve/veya sürekli değişim gösterdiği bölgelerde daha büyük genlikli olduğunu görülmektedir. Bu veriler, geribeslemeli kontrol sisteminin çalışması sırasında, filtre eklenmemiş durumda mıknatıs topun filtre eklenmiş duruma göre daha sarsıntılı bir şekilde havada asılı kaldığı gözlemini doğrulamaktadır.



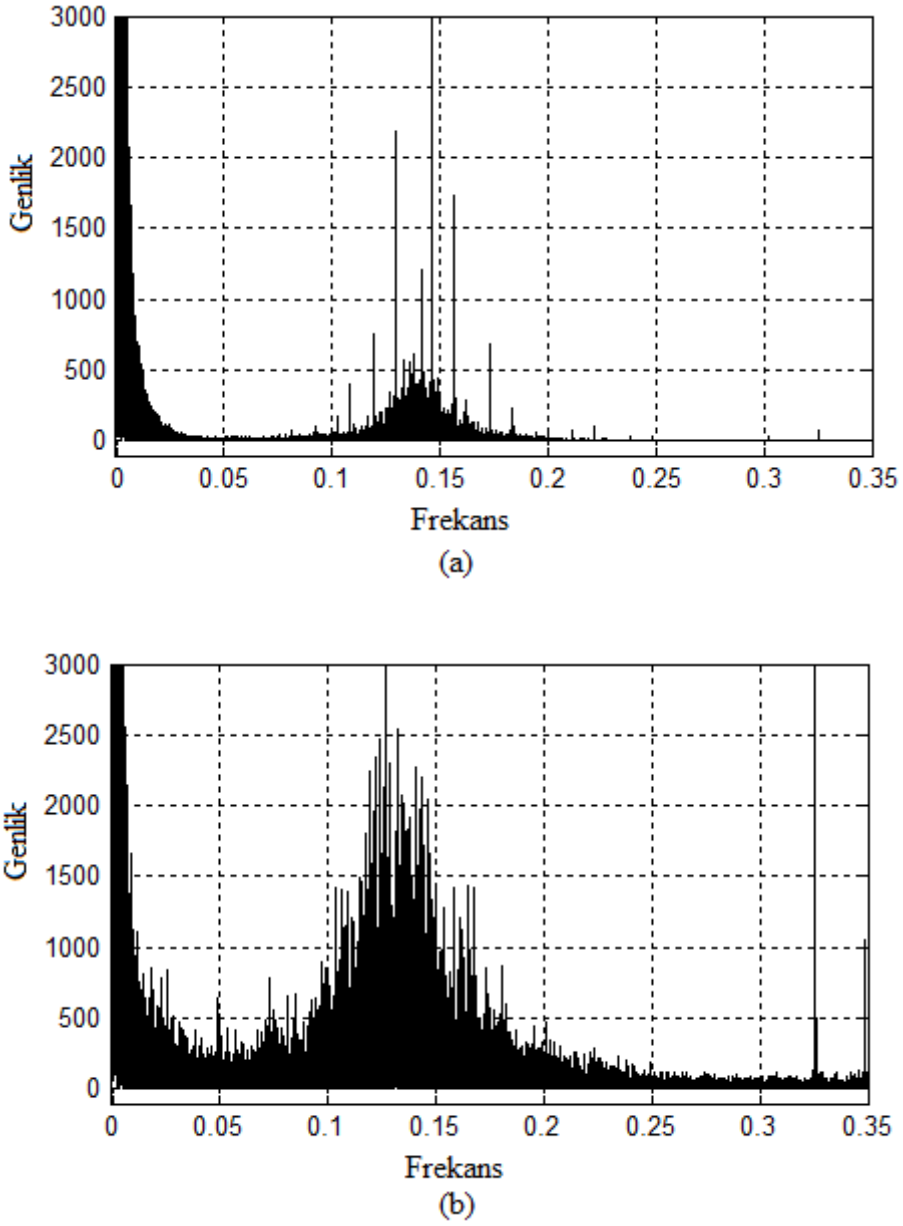
(a)



Şekil 6.3: Filtreleme işlemi eklenmiş geribeslemeli kontrol sistemine ait veriler: (a) Referans ve çıkış işareti, (b) Kontrol işareti ve durum değişkenleri

Şekil 6.4a'da filtrelenmiş durum ve Şekil 6.4b'de filtrelenmemiş durum olmak üzere sistem çıkış işaretine ait Fast Fourier Transform (FFT) grafikleri görülmektedir. Uygulanan referans işaretinin takibi dolayısıyla çıkış işareti temelde bir sabit/basamak ve bir sinüzoidal bileşen içermektedir. Grafiklerde solda bulunan büyük genlikli bölüm sabit/basamak bileşenine ait FFT ve ortaya yakın bulunan düşük genlikli bölüm de sinüzoidal bileşenine ait FFT sonuçlarını göstermektedir. Şekil 6.4a'da her iki bölümün de frekans ekseninde dar bir alana sıkıştırıldığı ve diğer bölgelerde bulunan frekans bileşenlerinin genliklerinin bastırılarak, ihmal edilebilecek seviyelere indirildiği görülmektedir. Buna rağmen Şekil 6.4b'de bu bölümlerin birbirlerinin içine girecek şekilde yayılmış olduğu ve

baskın frekans bölümleri haricinde bulunan frekans bileşenlerinin dahi yeterince bastırılamadığı görülmektedir. Bu durum filtreleme işlemi ile yüksek frekanslı gürültü bileşenlerinin başarılı bir şekilde bastırıldığını göstermektedir.

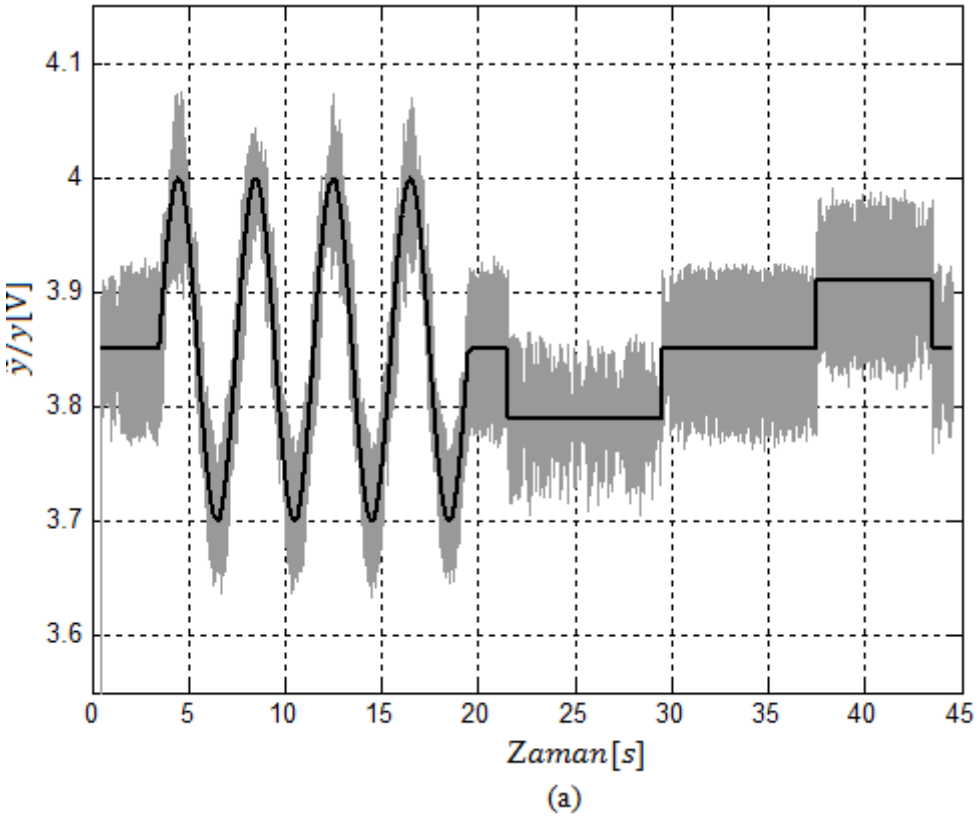


Şekil 6.4: Çıkış işareti FFT grafikleri: (a) Filtrelenmiş, (b) Filtrelenmemiş

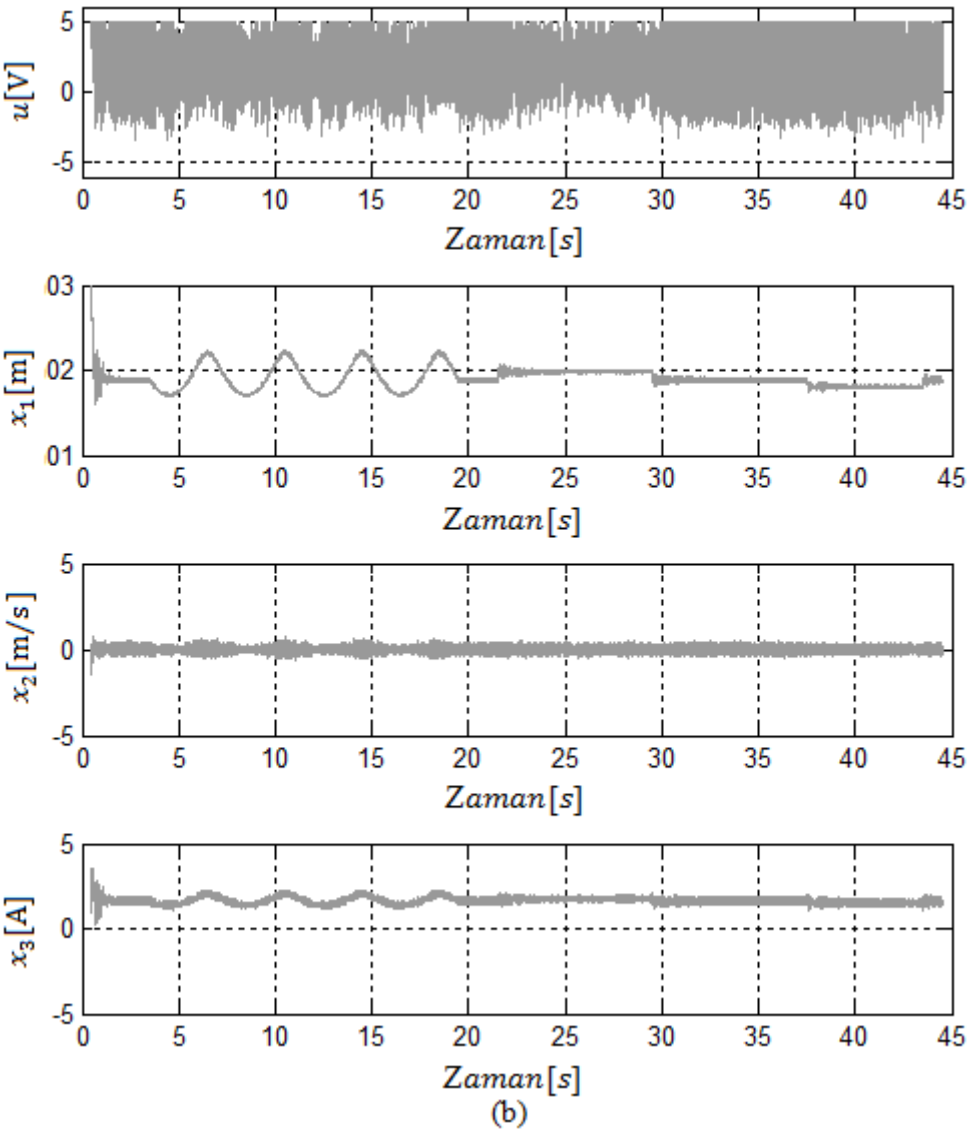
Yapılan gerçek-zamanlı deneyler sırasında RKMPC yönteminin kabul edilebilir bir performans sağlayabilmesi için, (2.31)'de λ simgesi ile verilen giriş işaretlerindeki değişimi cezalandıran parametrenin olabildiğince küçük olması gerektiği sonucuna varılmış ve $\lambda = 0.0001$ değerinde kullanılmıştır. Bununla

birlikte $T_s \times H$ şeklinde gösterilebilecek olan kestirim ufku süresinin yaklaşık olarak 2-7 ms aralığında kalması gerektiği tespit edilmiştir.

2 ms'den daha kısa kestirim ufku kullanımında, küre mıknatısın keskin bir şekilde aşağı yukarı hareketler sergilediği görülmüştür. Bunun sebebi yeterince uzağa bakılmadığı için sistem çıkışının ulaşacağı değerin tam olarak belirlenememesi ve ölçüm sırasında bu değerle karşılaşıldığında da ihtiyacı karşılamak için sert ve güçlü tepki vermek zorunda kalınmasıdır. Şekil 6.5'te kısa kestirim ufku için verilmiş olan grafiklerden hem kontrol işaretinin keskin ve büyük değerlikli değişimleri, hem de bunun sonucunda küre mıknatısın sarsıntılı bir şekilde havada asılı kaldığı anlaşılmaktadır. Bununla birlikte referans işaretinin keskin değişimlerinde sistem çıkışının referans işaretine ulaşmak için bir süre salınım yapmakta olduğu Şekil 6.5b'deki x_1 ve x_2'ye ait alt grafiklerde görülmektedir.



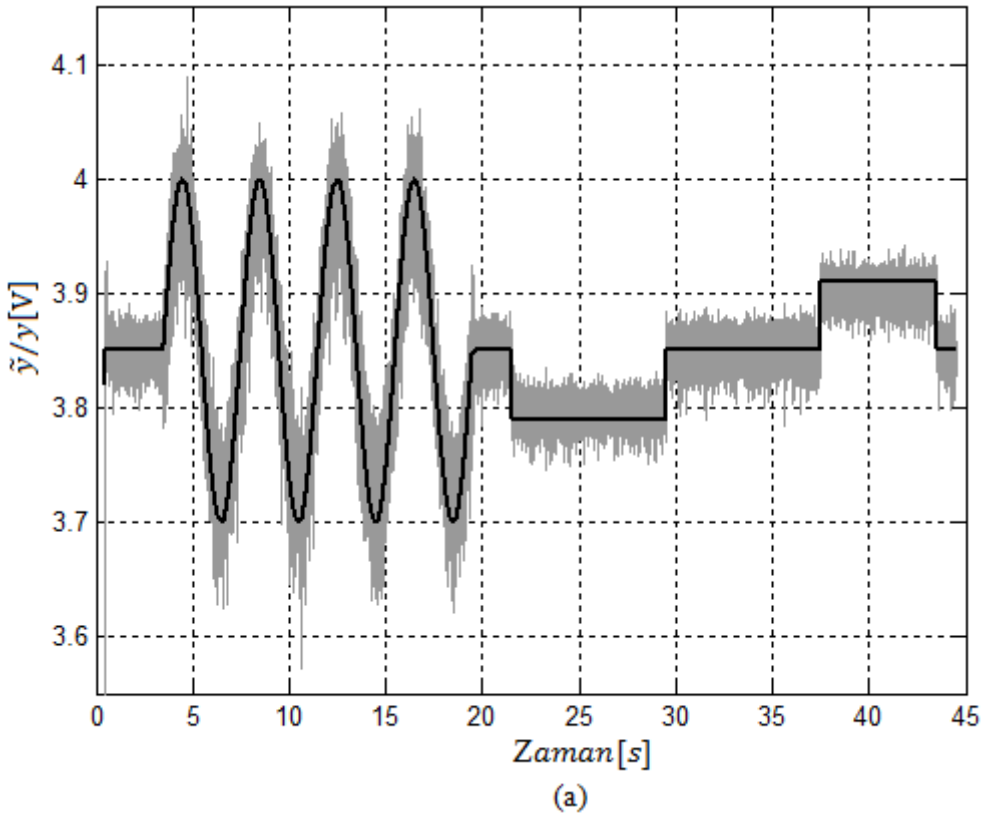
(a)



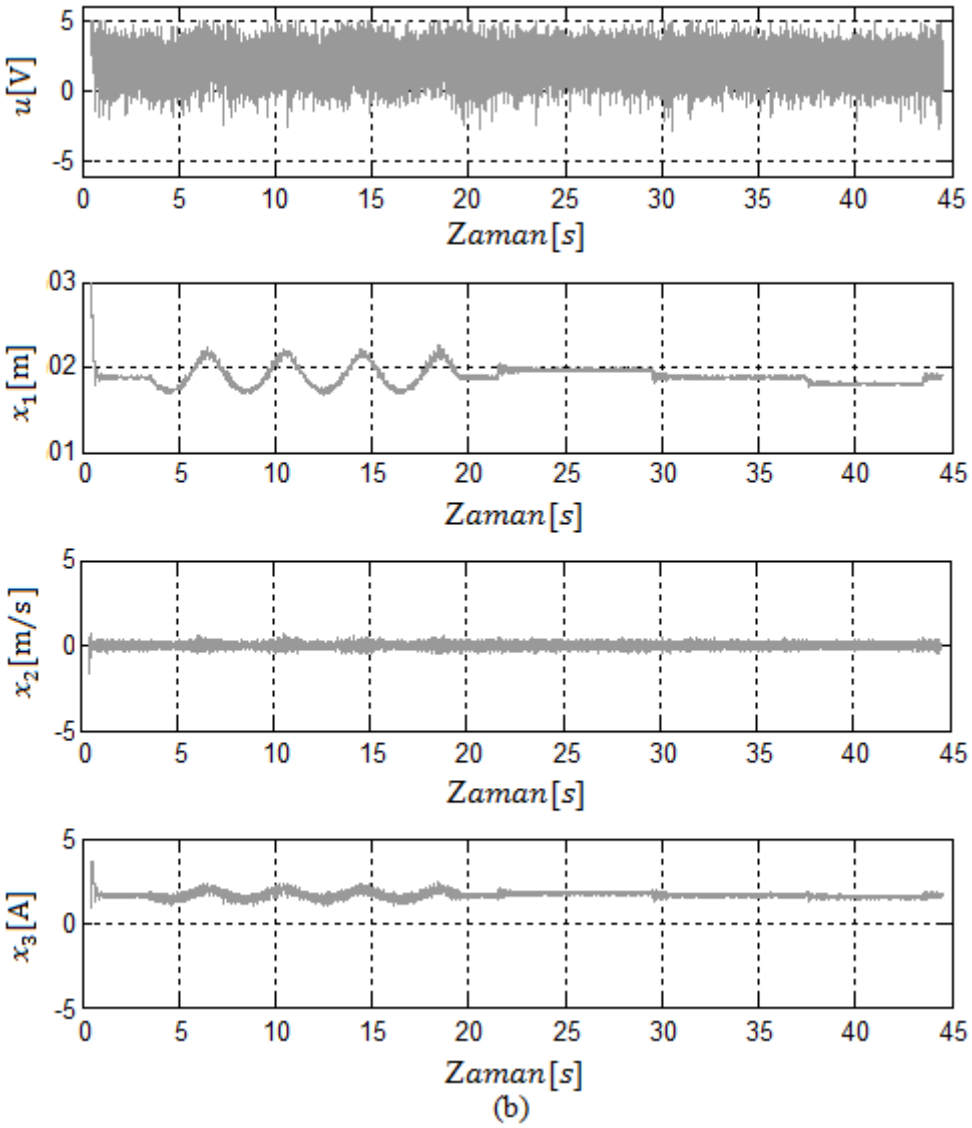
Şekil 6.5: Kısa kestirim ufuklu (H=4) geribeslemeli kontrol sistemi verileri: (a) Referans ve çıkış işareti, (b) Kontrol işareti ve durum değişkenleri

Şekil 6.6'da uzun kestirim ufku kullanımına ait veriler görülmektedir. Bu grafikte özellikle Şekil 6.6b'deki kontrol işaretinin yumuşak ve limit noktalardan uzak bir şekilde değişimi ve Şekil 6.6b'deki x_1 alt grafiği üzerinde sinüzoidal referans işaretinin takibi sırasında görülen dalgalanmalar dikkati çekmektedir. Burada kontrol işaretinin yavaş değişimi ve yetersiz kalışı açıkça görülmektedir. 7 ms'den daha fazla süreli kestirim ufku kullanımında ise küre mıknatısın zeminden kaldırılamadığı görülmüştür. Çünkü, uzak mesafelere bakıldığı için kontrolör olması gerektiğinden daha temkinli davranmakta ve bu durumda da yeterli performansı sağlayacak güçte bir kontrol işareti üretememektedir.

Kestirim ufku uzunluğunun belirlenmesine yönelik bu deneylerin sonucunda en doğru kontrol işaretini bulabilmek için en uygun kestirim ufku uzunluğunun 3-4 ms olması gerektiği sonucuna varılmıştır.



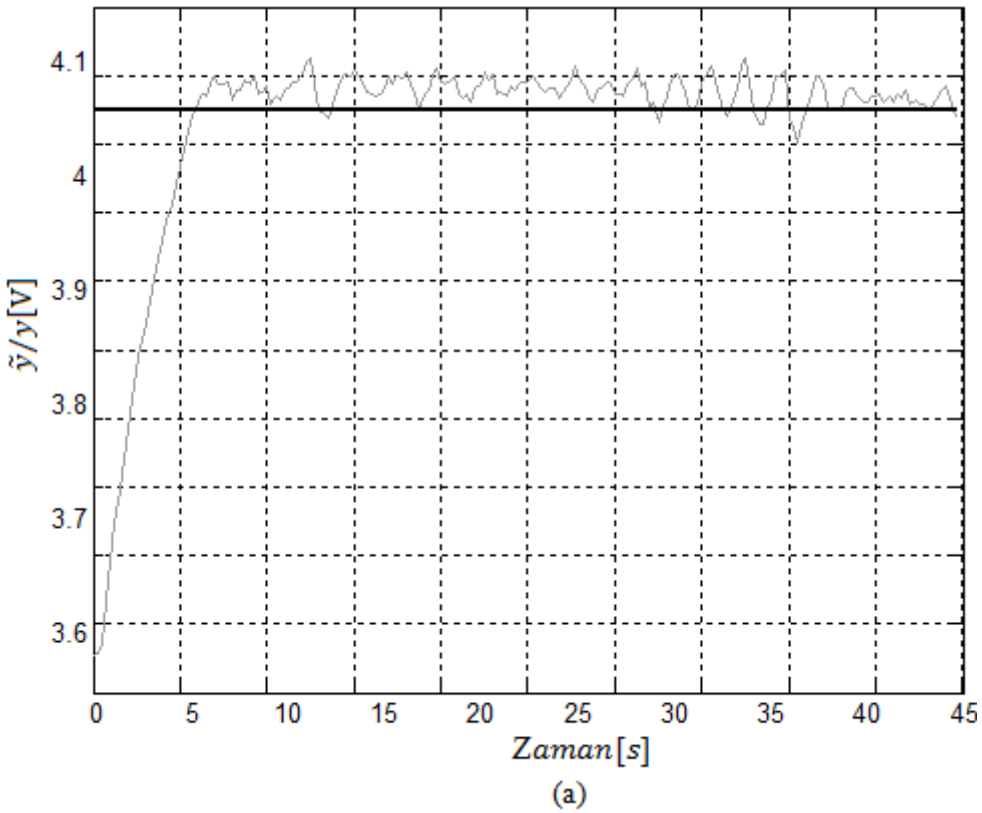
(a)



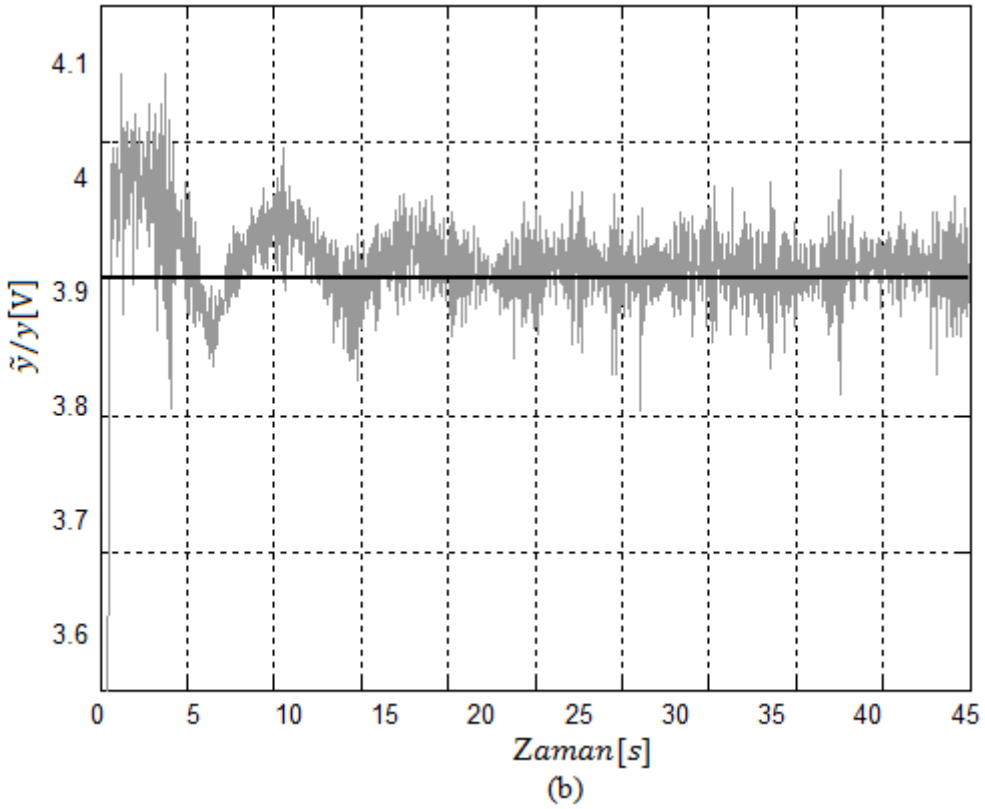
Şekil 6.6: Uzun kestirim ufuklu (H=9) geribeslemeli kontrol sistemi verileri: (a) Referans ve çıkış işareti, (b) Kontrol işareti ve durum değişkenleri

Kontrolörün performansının değerlendirilmesi için sistemin Geçici-Hal ve Sürekli-Hal davranışlarının da incelenmesi gerekir. Sistemin Geçici-Hal davranışlarını belirlemek için kontrol eyleminin başlatılmasının ardından geçen 3 s'lik zaman dilimi incelenmiş ve bu bölüme ait veriler Şekil 6.7'de verilmiştir. Şekil 6.7a'da Geçici-Hal davranışının sergilendiği ilk bölüme odaklanılmış ve burada sistemin yükselme zamanı bilgisi elde edilmiştir. Buna göre sistemin yükselme zamanı yaklaşık olarak 11 ms'dir ve örnekleme zamanının $T_s = 0.5$ ms değerinde olduğu düşünülürse bu süre 22 örnekleme adımına karşılık gelmektedir. Bu durum kontrolörün oldukça hızlı davrandığını ve sistem çıkışının çok kısa bir sürede 0 seviyesinden referans işareti seviyesine ulaştığını göstermektedir. Şekil

6.7b'de ise sistemin Geçici-Hal davranışının gerçekleştiği 1-2 s aralığındaki zaman dilimine ait veriler görülmektedir. Burada sistem çıkışının başlangıç anından sonra referans noktasından yaklaşık olarak 0.05 V yukarıya ulaştığı ve sonrasında salınıma başladığı görülmektedir. Taban seviyesi yaklaşık 3.1 V ve ayar noktası 3.85 V olduğuna göre bu durum maksimum aşım değerinin yaklaşık %6 olduğunu göstermektedir. Şekil 6.7'de ayrıca sistem çıkışının oturma zamanının da yaklaşık 1.6 s olduğu görülmektedir. Geçici-Hal davranışının sergilendiği bu bölümde küre mıknatıs zeminde bulunduğu noktadan kaldırılmakta ve en kısa sürede ve en az maksimum aşım değeri ile referans noktası üzerinde asılı kalması sağlanmaktadır. Bu durum Doğrusal Olmayan kontrol problemi olup, RKMPC yönteminin FPGA üzerinde gerçeklenmesi ile elde edilmiş kontrolör bunu başarıyla gerçekleştirmiştir.

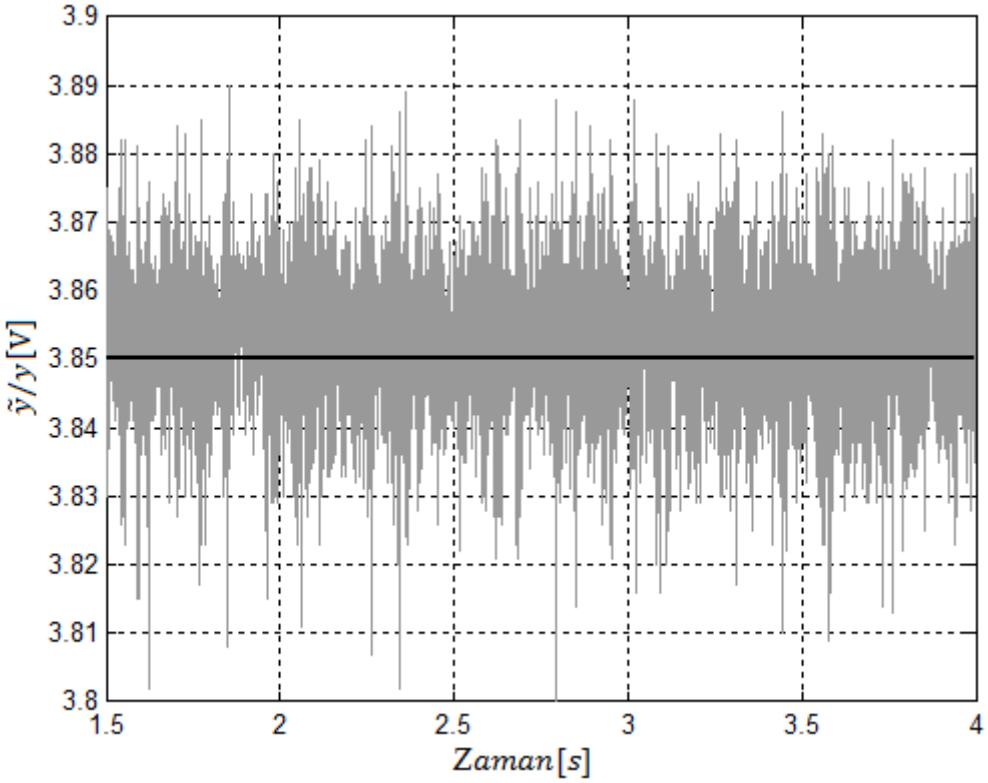


(a)



Şekil 6.7: Geçici-Hal davranışı: (a) Kısa (b) Uzun

Şekil 6.8'de sistemin Sürekli-Hal davranışı detaylı olarak gösterilmiştir. Bu grafikte anlık gürültü oldukları apaçık belli olan veriler dikkate alınmadığında, sistemin Sürekli-Hal durumunda 3.82-3.88 V aralığında salındığı görülmektedir. Bu durumda, sistem çıkış geriliminin en küçük 3.1 V ve en büyük 4.5 V değerine ulaştığı dikkate alındığında, salınım miktarının yaklaşık %5 civarında olduğu sonucuna ulaşılır. Bu salınım miktarı yaklaşık 0.9 mm'lik bir aralığa karşılık gelmektedir. Bu durum, deneyler yapılırken kontrol eylemi sırasında küre mıknatısın havada neredeyse hareketsiz bir şekilde durduğu şeklindeki gözlemleri doğrulamaktadır. Referans işareti, EMLS'nin denge noktası etrafında olacak şekilde seçildiğinden, küre mıknatısın zeminde bulunduğu yerden havaya kaldırılarak referans noktası etrafında salınım yapar hale gelmesinden sonraki süreç bir doğrusal kontrol problemidir ve RKMPC yöntemine dayalı olarak oluşturulan gömülü kontrolör başarılı bir şekilde bu problemin de üstesinden gelmiştir.



Şekil 6.8: Sürekli-Hal davranışı

Geribeslemeli kontrol sistemi sabit, sinüzoidal ve basamak olmak üzere 3 farklı referans işaretini takip edebilmektedir. Sinüzoidal referans işareti,

$$\tilde{y}[n] = \mathrm{A} + \mathrm{V_{p1}} \times \sin(0.5\pi n) \tag{6.1}$$

şeklinde ve basamak işareti,

$$\tilde{y}[n] = \begin{cases} \mathrm{A} - \mathrm{V_{p2}} & 0 \le n \le T1 \\ \mathrm{A} & T1 \le n \le T2 \\ \mathrm{A} + \mathrm{V_{p2}} & T2 \le n \le T3 \end{cases} \tag{6.2}$$

şeklindedir. Burada A, sabit referans değerini (3.85 V), V_{p1}, sinüzoidal işaretin tepe değerini (0.15 V), V_{p2} de basamak değerini (0.08 V) göstermektedir ve bu değerler DE2-115 üzerindeki iki konumlu ve çıt-çıt anahtarlar kullanılarak ayarlanabilmekte ve ayarlanan değer nümerik göstergeler üzerinde görüntülenmektedir. Deneysel çalışmalar sırasında öncelikle sabit referans takibi daha sonra basamak referans takibi ve son olarak sinüzoidal referans takibi sağlanmıştır. Bu referans tipleri önce, kontrol eylemi sırasında DE2-115

üzerindeki iki konumlu anahtarlar kullanılarak değiştirilmiş fakat daha sonra, geribeslemeli kontrol sisteminden alınan verilerin standart süreler içermesi için, denklem (6.2)'de görüldüğü gibi sabit biçimli bir referans işareti olarak kullanılmıştır. Bu şekilde farklı çalışma tipleri için alınan verilerin birbiri ile karşılaştırılması daha kolay hale gelmiştir.

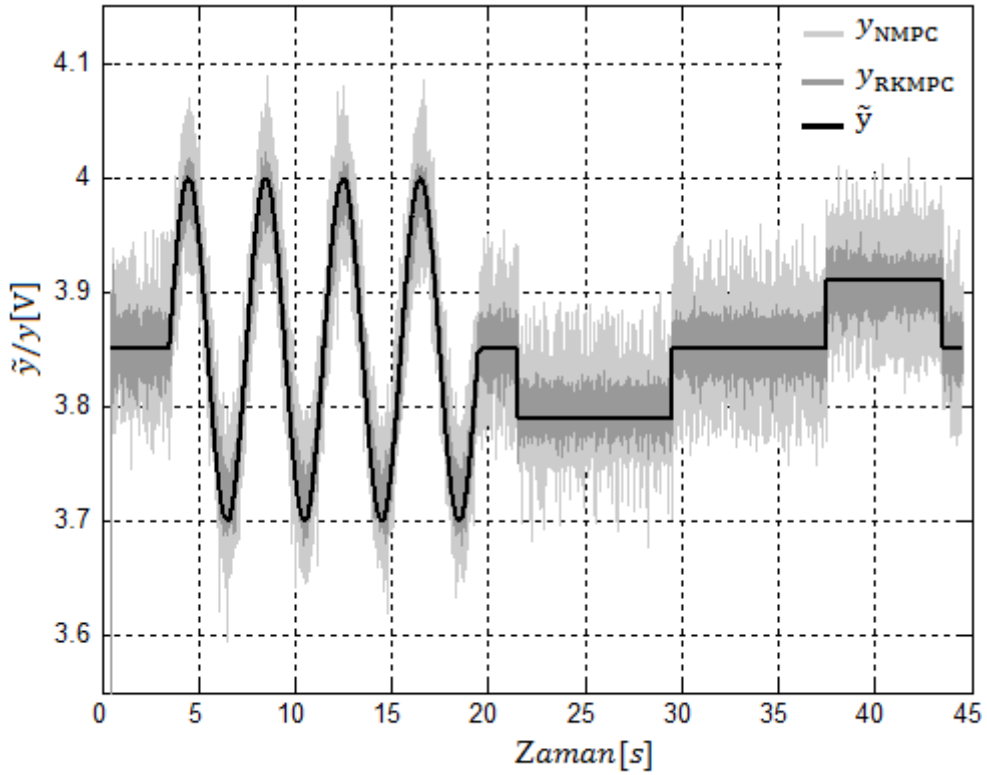
$$\tilde{y}[n] = \begin{cases} \text{PWM\_OFF} & 0 \leq \text{n} \leq 2000 \quad (0 - 1\ \text{s}) \\ \text{Sabit Ref.} & 2000 \leq \text{n} \leq 8000 \quad (1 - 4\ \text{s}) \\ \text{Sinüzoidal Ref.} & 8000 \leq \text{n} \leq 40000\ (4 - 20\ \text{s}) \\ \text{Sabit Ref.} & 40000 \leq \text{n} \leq 44000\ (20 - 22\ \text{s}) \\ \text{Basamak Ref.} & 44000 \leq \text{n} \leq 88000\ (22 - 44\ \text{s}) \\ \text{Sabit Ref.} & 88000 \leq \text{n} \leq 90000\ \ (44 - 45\ \text{s}) \\ \text{PWM\_OFF} & 90000 \leq \text{n} \qquad (45\text{s} - \quad) \end{cases} \tag{6.3}$$

Sonuç olarak $T_s = 0.5$ ms şeklindeki örnekleme zamanı için kestirim ufku büyüklüğü $H = 8$ ve algılayıcı çıkışlarında filtreleme işlemi amacıyla ortalaması alınacak değer adedi 9 seçildiğinde, FPGA üzerinde RKMPC algoritmasının gerçeklenmesi ile elde edilen kontrolörün EMLS'yi başarıyla kontrol etmesi sağlanmıştır.

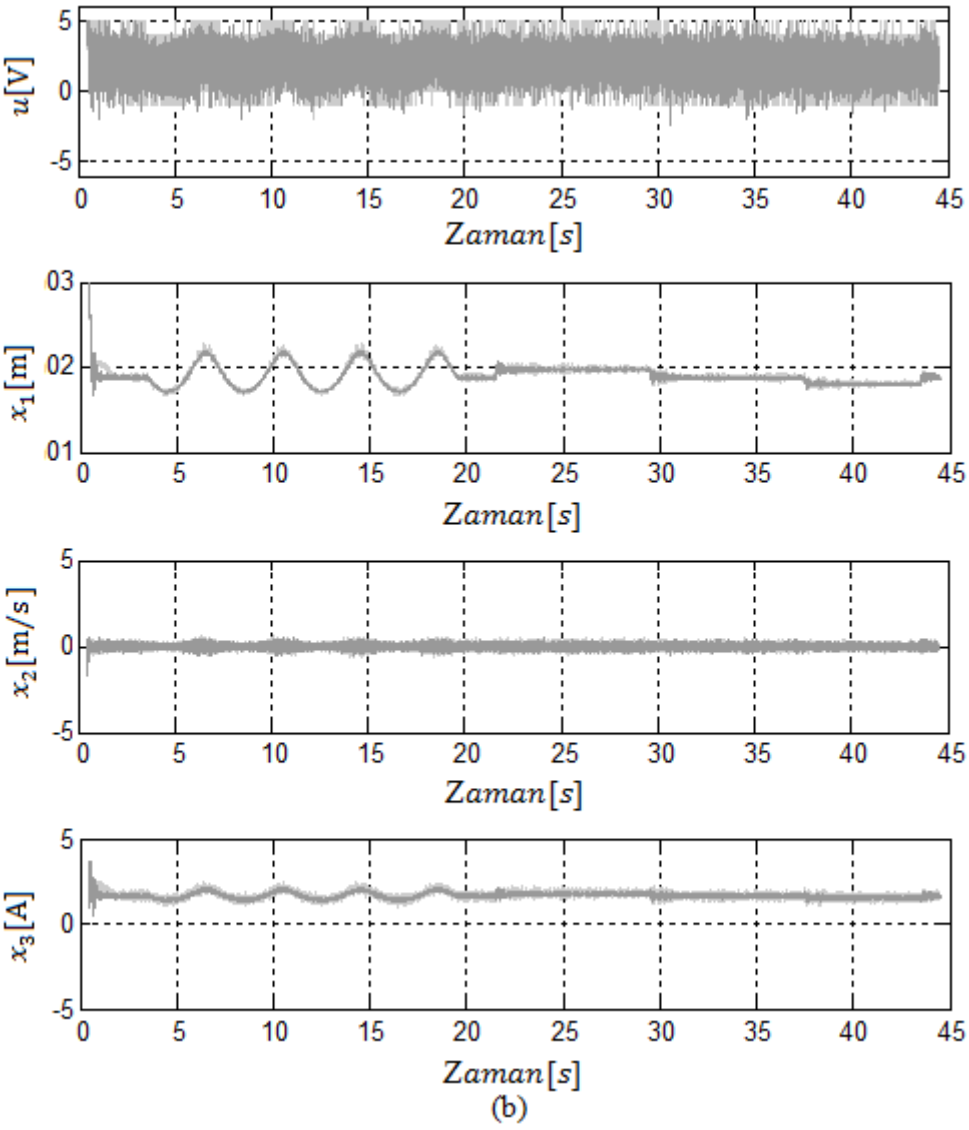
6.2. FPGA Üzerinde RKMPC ve NMPC Yöntemlerinin Karşılaştırılması

FPGA üzerinde RKMPC algoritmasının gerçeklenmesi ile elde edilen kontrolörün EMLS'yi kontrolü sırasında uygulanan işlemler, benzer şekilde NMPC yönteminin FPGA üzerinde gerçeklenmesi ile elde edilmiş olan kontrolör ile tekrarlanmıştır. NMPC yöntemi olarak çok çeşitli yöntemler önerilmiş olmasına rağmen, bu tez çalışmasında, Plucenio ve diğ. tarafından önerilen yöntem geleneksel Doğrusal Olmayan MPC (NMPC) yöntemi olarak kabul edilmiş (Plucenio ve diğ. 2007)ve FPGA üzerinde bu yönteme ait algoritma gerçeklenmiştir. Aynı referans işaretinin, her iki geribeslemeli kontrol sistemine

de aynı çalışma parametreleri altında uygulanması sonucunda elde edilen veriler Şekil 6.9'de verilmiştir.



(a)



Şekil 6.9: NMPC ve RKMPC yöntemlerinin karşılaştırılması: (a) Referans ve çıkış işareti, (b) Kontrol işareti ve durum değişkenleri

Geleneksel NMPC algoritması genel olarak RKMPC algoritmasına benzemekle birlikte, kestirim ufku boyunca sistem çıkışlarına ait türev bilgilerinin elde edilmesi noktasında farklılaşmaktadır. NMPC algoritmasında bu türev bilgilerinin elde edilmesi için her kestirim adımında sadece durum denklemlerinin türevlerinden $\left(\frac{\partial \hat{\mathbf{f}}}{\partial x}=\frac{\partial \mathbf{f}}{\partial x},\frac{\partial \hat{\mathbf{f}}}{\partial u}=\frac{\partial \mathbf{f}}{\partial u}\right)$ faydalanılırken, RKMPC algoritmasında RK parametreleri olan $\mathbf{k}$ parametrelerinin türevlerinden $\left(\frac{\partial \hat{\mathbf{k}}_i}{\partial x},\frac{\partial \hat{\mathbf{k}}_i}{\partial u}\right)$ ve RK modelinin türevlerinden $\left(\frac{\partial \hat{\mathbf{f}}}{\partial x},\frac{\partial \hat{\mathbf{f}}}{\partial u}\right)$ faydalanılmaktadır. Bu nedenle RKMPC yöntemi, daha hassas değerlere ulaşarak kontrol işareti üzerinde daha yumuşak değişimlerin oluşmasını sağlamakta ve bunun sonucunda daha başarılı bir kontrol

gerçekleştirilmiş olmaktadır. NMPC için ise durum bu kadar başarılı olmayıp, küre mıknatısın havada daha sarsıntılı bir şekilde asılı kaldığı gözle görülmekte ve bu durum Şekil 6.9 ile verilen grafiklerde kendini göstermektedir.

NMPC ve RKMPC gerçeklemelerine ilişkin olarak 3 farklı tipte referansa ait izleme hataları Tablo 6.2'de görülmektedir. Root Mean Square Error (RMSE) şeklinde elde edilmiş olan bu veriler incelendiğinde, RKMPC gerçeklemesinde sistem çıkışının her 3 tip referans işaretini de NMPC gerçeklemesine göre 5-6 kat daha yakından takip ettiği görülür.

Tablo 6.2: RKMPC ve NMPC gerçeklemelerinin RMSE performansları

Referans Tipi	**RKMPC**	**NMPC**
Sabit	1.7610	6.7570
Sinüzoidal	5.8951	26.3187
Basamak	5.1664	33.4144

Bu bilgiler doğrultusunda NMPC yönteminin FPGA üzerinde gerçeklenmesi ile elde edilen kontrolörün kabul edilebilir bir performans gösteriyor olmasına rağmen, RKMPC yönteminin FPGA üzerinde gerçeklenmesi ile elde edilen kontrolörün daha iyi bir performans sağladığı görülmektedir.

6.3. RK Tabanlı Parametre Kestirimi ile Model-Uyarlamalı Öngörülü Kontrol

RKMPC algoritmasının FPGA üzerinde gerçeklenmesiyle oluşturulan kontrolöre, RKPE eklenmesi ile EMLS'nin Model-Uyarlamalı Öngörülü Kontrol işlemi gerçekleştirilmiştir. RKMPC algoritması ile RKPE algoritmasının benzerliği Bölüm 5.6'da tüm detayları ile açıklanmış ve bu benzerlik dolayısıyla iki algoritma Şekil 5.17'de görüldüğü gibi bütünleşik olarak FPGA üzerinde gerçeklenmiştir. RKMPC algoritmasına eklenen $\frac{\partial \hat{\mathbf{k}}_i}{\partial \theta}$, $\frac{\partial \hat{\mathbf{f}}}{\partial \theta}$, $\frac{\partial \hat{\mathbf{x}}}{\partial \theta}$, $\frac{\mathbf{J}_\theta^T \mathbf{e}}{\mathbf{J}_\theta^T \mathbf{J}}$ ve $\theta[n+1]$ terimlerinin nasıl hesaplanacağına ilişkin denklemler de Bölüm 4.3'de verilmiştir. EMLS'nin sistem denklemlerinde bulunan parametrelerin değerleri, bir kısmı ölçülerek, bir kısmı da Bölüm 5.3'te verilen yönerge uygulanarak

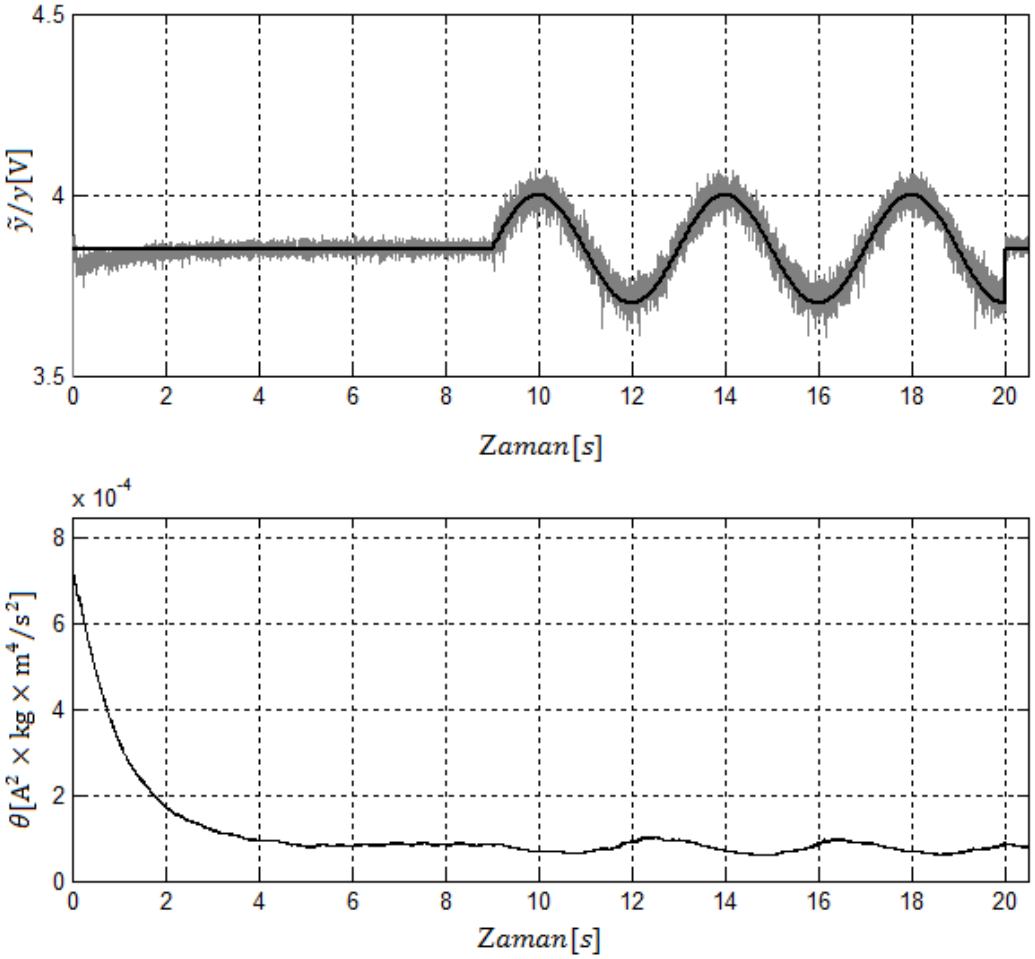
belirlenebilmektedir. Buna rağmen, kestirimi yapılacak olan parametre olarak önce, küre mıknatısın ağırlığını ifade eden *m* parametresi seçilmiş fakat sonra, bu parametrenin özellikle elektromanyetik güç katsayısını temsil eden *k* parametresine bağlı olarak değiştiği görülmüştür. Bölüm 5.3'te verilen yönergeye göre *k* değerinin bulunması için öncelikle, herhangi bir kontrolör kullanılarak, küre mıknatısın bobinden yaklaşık 2 cm uzaklıkta bulunan denge noktasında tutulması gerekmektedir. Bu sırada küre mıknatısa uygulanan yerçekimi kuvveti ile elektromanyetik kuvvetin birbirine eşitlendiği varsayılarak, (6.4) elde edilmiştir(Zeltom 2015).

$$m\ddot{d} = k\frac{v/R}{d^3} \tag{6.4}$$

Burada *v* parametresi bobine uygulanan giriş işaretini temsil eder ve kontrolör tarafından hesaplanır, d parametresi ise küre mıknatısın bobine olan uzaklığını ifade eder ve bobine uygulanan giriş işaretine göre değişmektedir. Dolayısıyla *m* parametresinin değerinin kestirimini yapabilmek için öncelikle *k* parametre değerinin bilinmesi gerekmektedir. Benzer şekilde *m* parametre değerinin bulunabilmesi için de *k* parametre değerinin bilinmesi gerekir. Bu nedenle kestirim yapılacak olan parametre olarak, aşağıdaki ifade seçilmiştir:

$$\theta = k/m \tag{6.5}$$

Bu parametrenin RK tabanlı çevrim-içi kestirimi yapılırken Model-Uyarlamalı Öngörülü Kontrol gerçekleştirilmiş ve elde edilen verilere ait grafik Şekil 6.10'da verilmiştir.



Şekil 6.10: Parametre kestirimi ile model-uyarlamalı öngörülü kontrol verileri

Şekil 6.10 incelendiğinde kontrol eyleminin başlatılmasından yaklaşık 5 s sonra parametrenin değerinin sabit bir değere yakınsadığı görülür. Bu ana kadar sistem çıkış işaretinin, referans işaretini yeterince doğru bir şekilde takip edemediği fakat daha sonra bunun sağlandığı görülmemektedir. Sinüzoidal referans işareti uygulanması sırasında çıkış işaretindeki dalgalanmanın arttığı fakat parametre değerinde oluşan değişikliğin ihmal edilebilir bir seviyede kaldığı görülmektedir.

7. SONUÇ VE ÖNERİLER

Bu tez çalışmasında, Altera firmasının DE2-115 FPGA eğitim ve geliştirme seti üzerinde bulunan Cyclone IV tip FPGA üzerinde RKMPC yöntemi (İplikçi 2013) gerçeklenmiştir. Tasarlanan kontrolör kullanılarak, kararsız, çok hızlı ve doğrusal olmayan bir sistem olan EMLS'nin kontrolü sağlanmıştır, Kontrolör ile EMLS arasında, üzerinde sistemden veri toplamak amacıyla iki kanal ADC ve üretilen kontrol işaretini sisteme aktarmak için bir bobin sürücü devresi bulunan ek kart tasarlanmıştır. Daha sonra, geleneksel Doğrusal Olmayan MPC olarak kabul edilen ve (Plucenio ve diğ. 2007)'de önerilen yöntemi uygulayan algoritma FPGA üzerinde gerçeklenerek, RKMPC gerçeklemesi ile ilgili çalışmalar tekrarlanmıştır. Böylece çalışma sırasında elde edilen veriler kullanılarak, her iki yönteme ait FPGA gerçeklemeleri ile oluşturulan iki farklı geribeslemeli kontrol sisteminin birbirleri ile karşılaştırılması sağlanmıştır. Son olarak, RKPE algoritması (İplikçi 2013), RKMPC algoritması ile bütünleşik olarak FPGA üzerinde gerçeklenmiş ve $\theta = {}^{k}/_{m}$ şeklinde seçilen parametrenin değerinin kestirimi sırasında EMLS'nin Model-Uyarlamalı Öngörülü Kontrol işlemi gerçekleştirilmiştir.

RKMPC, NMPC ve RKPE algoritmaları önce MATLAB/Simulink üzerinde gerçeklenerek Hilink kontrol kartı üzerinde EMLS'nin kontrolü sağlanmıştır. Daha sonra, doğruluğu bu şekilde onaylanan algoritmaların FPGA üzerinde donanımsal olarak gerçeklenmesi için gerekli yazılım, Altera firmasının Quartus II programının 13.1d Web Sürümü üzerinde, VHDL kodlama tekniği kullanılarak gerçekleştirilmiştir. Bu VHDL kodlarının, ModelSim Altera programı üzerinde benzetimleri yapılmış ve tüm hesaplama işlemlerinde elde edilen sonuçların MATLAB üzerinde yapılan hesaplamalarla aynı değerlerde olduğu görüldükten sonra donanım FPGA üzerine gömülmüştür. Temel modüller 32-bit floating point yapısındaki MegaFunction araçları olmak üzere, tüm donanım, FSM programlama yapıları ile oluşturulmuş ve birbirine ardışık düzen (pipelined) olarak bağlanmış olan pek çok modül kullanılarak gerçeklenmiştir. Algoritmaların sıralı yapısı nedeniyle bazı işlemler sıralı yapılmak zorunda kalınsa da, birbirinden bağımsız durumda bulunan işlemler, FPGA kaynaklarının izin verdiği ölçüde paralel olarak gerçekleştirilmiştir. Yazılan VHDL kodlarının doğruluğu

ModelSim Altera benzetim programı kullanılarak test edildikten sonra FPGA üzerine yüklenmiştir. Oluşturulan geribeslemeli kontrol sistemleri üzerinden verilerin toplanarak bilgisayara aktarılması için Quartus II programı içinde bütünleşik olarak bulunan SignalTap II Logic Analyzer aracı kullanılmıştır. Böylece, $T_s = 0.5\text{ms}$ olan örnekleme periyotlarının her birinde geribeslemeli kontrol sistemi üzerinden durum değişkenlerinin, çıkış işaretinin, kontrol işaretinin, referans işaretinin ve kestirimi yapılan parametrenin değerlerinin bilgisayar tarafında elde edilmesi sağlanmıştır. Bu verilerin MATLAB kullanılarak grafikleri oluşturulmuş ve değerlendirmeleri yapılmıştır.

Elde edilen bulguların değerlendirilmesi sonucunda, RKMPC yönteminin FPGA üzerinde gerçeklenmesi ile elde edilen kontrolörün, hem çıkış işaretinin referans işaretini izlemesi, hem kontrol işaretinin yeterliliği, hem de durum değişkenlerinin değerleri açısından oldukça başarılı bir kontrol performansı sergilendiği görülmüştür. Buna karşılık NMPC yönteminin FPGA gerçeklemesi ile elde edilen kontrolörün de kabul edilebilir bir kontrol performansı göstermesine rağmen, RKMPC gerçeklemesinin daha üstün bir kontrol performansı gerçekleştirdiği ortaya konmuştur. Son olarak RKMPC algoritmasına RKPE algoritması eklenerek oluşturulan RKMPC/RKPE algoritmasının FPGA üzerinde gerçeklenmesi ile elde edilen kontrolör kullanılarak $\theta = {}^{k}/_{m}$ şeklinde seçilen bir parametrenin çevrim-içi kestiriminin yapılması sırasında EMLS'nin kontrol edilmesi ile Model-Uyarlamalı Öngörülü Kontrol işlemi başarılı bir şekilde gerçekleştirilmiştir.

8. KAYNAKLAR

Allgöwer, F., Findeisen, R., and Nagy, Z. K.," Nonlinear Model Predictive Control:From Theory to Application*", J. Chin. Inst. Chem. Engrs.*, 35(3), 299-315, (2004).

Allegro, "A1302 - Continuous-Time Ratiometric Linear Hall Effect Sensor ICs", (19/03/2015), http://www.allegromicro.com/~/media/Files/Datasheets/A1301-2-Datasheet.ashx, (2015a).

Allegro, "ACS715 - Automotive Grade, Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 2.1 kV RMS Voltage Isolation and a Low-Resistance Current Conductor", (19/03/2015), http://www.allegromicro.com/~/media/Files/Datasheets/ACS715-datasheet.ashx, (2015b).

Altera, "DE2-115 User Manuel", (19/05/2015), http://www.terasic.com.tw/cgi-bin/page/archive_download.pl?Language=English&No=502&FID=cd9c7c1feaa2467c58c9aa4cc02131af, (2015).

Bleris, L.G., Kothare, M.V., "Implementation of Model Predictive Control for Glucose Regulation on a General Purpose Microprocessor", *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, 5162-5167, (2005).

Bleris, L.G.,Vouzis, P.D., Arnold, M.G., Kothare, M.V., "A co-processor FPGA platform for the implementation of real-time model predictive control", *American Control Conference*, 1912-1917, (2006).

Camacho E.F., "Constrained generalized predictive control", *IEEE Transactions on Automatic Control*, 38, 327-332, (1993).

Camacho E.F., and Bordons, C., *Model Predictive Control*, London: Springer-Verlag, (2007a).

Camacho E.F., and Bordons. C., "Nonlinear model predictive control: an introductory review", *Lecture Notes in Control and Information Sciences* 358, 1-16, (2007b).

Cheng, D. K., *Field and Wave Electromagnetics*, MA: Addison-Wesley, (1983).

Clarke D. W., Mohtadi, C. and Tuffs, P. C., "Generalized predictive control part 1: the basic algorithm", *Automatica*, 23, 137-148, (1987a).

Clarke D. W., Mohtadi, C. and Tuffs, P. C., "Generalized predictive control part 2: extensions and interpretations", *Automatica*, 23, 2, 149-163, (1987b).

Clarke D. W., Mohtadi, C., "Properties of generalized predictive control", *Automatica*, 25(6), 859-875, (1989).

Cutler, C.R., and Ramaker, B.L., "Dynamic matrix control: a computer control algorithm", *In Proceedings of the Joint Automatic Control Conference*, 17, San Francisco, U.S.A, 72, (1980).

De Keyser, R.M.C. and Van Cauwenberghe, A.R., "Extended prediction self-adaptive control", *In Proceedings of the 7th IFAC Symposium on Identification and System Parameter Estimation*, York, U.K., (1985).

Demircioglu, H. and Gawthrop, P.J., "Continuous-time generalized predictive control (CGPC)", *Automatica*, 27(1), 55-74, (1991).

Fairchild Semiconductor, "FDD8424 - Dual N and P-channel enhancement mode power MOSFET", (19/03/20015), https://www.fairchildsemi.com/ datasheets/FD/FDD8424H.pdf, (2015).

Garcia C. E., Prett, D. M., and Morari, M., "Model predictive control: Theory and practice - a survey", *Automatica*, 25(3), 335- 348, (1989).

Gawthrop, P.J. and Demircioglu, H., "Continuous-time generalized predictive control", *Proceedings of the IFAC Symposium on Adaptive Systems in Control and Signal Processing, Glasgow*, 123-128, (1989).

Hassapis, G., "Implementation of model predictive control using real-time multiprocessing computing", *Microprocessors and Microsystems*, 27(7), 327-340, (2003).

Henson, M.A., "Nonlinear model predictive control: current status and future directions", *Computers and Chemical Engineering*, 23(2), 187-202, (1998).

İplikçi S., "Support vector machines-based generalized predictive control", *International Journal of Robust and Nonlinear Control*, 16(17), 843-862, (2006).

İplikçi S., "A support vector machines-based control application to the experimental three-tank system", *ISA Transactions*, 49(3), 376-386, (2010).

İplikçi S., "Runge-kutta model-based adaptive predictive control mechanism for non-linear processes", *Transactions of the Institute of Measurement and Control*, 35(2), 166-180, (2013).

Jerez, J.L., Goulart, P.J., Richter, S., Constantinides, G.A., Kerrigan, E.C., and Morari, M., "Embedded Predictive Control on an FPGA using the Fast Gradient Method", *2013 European Control Conference (ECC)*, Zürich, Switzerland, 3614-3620, (2013).

Kawathekar, R. and Riggs, J., "Nonlinear model predictive control of a reactive distillation column", *Control Engineering Practice*, 15(2), 231-239, (2007).

Klatt, K. U. and Marquadt, W., "Perspectives for process systems engineering-personal views from academia and industry", *Computers and Chemical Engineering*, 33(3), 536-550, (2009).

Lau, M.S.K, Yue, S.P., Ling, K.V. and Maciejowski J.M., "A Comparison of Interior Point and Active Set Methods for FPGA Implementation of Model Predictive Control", *Proceedings of the European Control Conference 2009*, Budapest, Hungary, 156-161, (August 2009).

Linear Technology, "LTC1606 - General Purpose SAR ADC", (19/03/20015), http://www.linear.com/docs/1737, (2000).

Ling, K.V., Yue, S.P. and Maciejowski J.M., "A FPGA Implementation of Model Predictive Control", *Proceedings of the 2006 American Control Conference Minneapolis*, Minnesota, USA, 1930-1935, (June 2006).

Ling, K.V., Wu, B.F. and Maciejowski J.M., "Embedded Model Predictive Control (MPC) using a FPGA", *Proceedings of the 17th World Congress The International Federation of Automatic Control*, Seoul, Korea, 15250-15255, (July 2008).

Lu, Y., Li, D., Xu, Z., and Xi Y.,"Convergence Analysis and Digital Implementation of a Discrete-Time Neural Network for Model Predictive Control", *IEEE Transactions Industrial Electronics*, 61(12), 7035-7045, (2014).

Maciejowski J.M., *Predictive Control with Constraints*, Essex: Pearson Education Limited, (2002).

Microchip, "MCP14E5 - Dual High-Speed Power MOSFET Drivers With Enable", (19/03/20015), http://ww1.microchip.com/downloads/en/DeviceDoc/22062b.pdf, (2008).

Morari, M. and Lee, J. H., "Model predictive control: past, present and future", *Computers and Chemical Engineering*, 23(4-5), 667-682, (1999).

Morari, M. and Baric, M., "Recent developments in the control of constrained hybrid systems", *Computers and Chemical Engineering*, 30(10-12), 1619-1631, (2006).

Naouar, M., Naassani, A., Monmasson, E. and Slama-Belkhodja, I., "Fpga-based predictive current controller for synchronous machine speed drive", Power Electronics, *IEEE Transactions*, 23(4), 2115-2126, (2008).

Nocedal, J. and Wright, S.J., *Numerical Optimization*, New York: Springer, (1999).

Plucenio, A., Pagano, D. J., Bruciapaglia, A. H., Normey-Rico, J. E.," A Practical Approach to Predictive Control for Nonlinear Processes", *7th IFAC Symposium on Nonlinear Control Systems*, 7(1), South Africa, 210-215, (2007).

Qin, S.J., and Badgwell, T.A., "A survey of industrial model predictive control technology", *Control Engineering Practice*, 11(7), 733-764, (2003).

Richalet J.A., Rault, A., Testud, J.L. and Papon, J., "Model predictive heuristic control: applications to an industrial process", *Automatica*, 14(5), 413-428, (1978).

Sandoz, D.J., Thorpe, P.J., Kurth, T., Desforges, M.J., Woolley, I.S., "Innovation in industrial model predictive control", *Model Predictive Control: Techniques and Applications - Day 2 (Ref. No. 1999/096), IEE Two-Day Workshop on*, 10(5), 189-197, (1999).

Schafer, A., Kuhl, P., Diehl, M., Schloder, J. and Bock, H., "Fast reduced multiple shooting methods for nonlinear model predictive control", *Chemical Engineering and Processing*, 46(11), 1200-1214, (2007).

Scokaert, P.O.M., Mayne, D.Q. and Rawlings, J.B., “Suboptimal model predictive controllers (feasibility implies stability)”, *IEEE Transactions on Automatic Control*, 44(3), 648–654, (1999).

Sira-Ramirez, H., "A Geometric Approach to Pulse-Width Modulated Control in Nonlinear Dynamic Systems", *IEEE Transactions on Automatic Control*, 34(3), 184-187, (1989).

Sira-Ramirez, H., Linschinsky-Arenas, P., "Dynamical Discontinuous Feedback Control of Nonlinear Systems", *IEEE Transactions on Automatic Control*, 35(12), 1373-1378, (1990).

Sistu, P. and Bequette, B., “Nonlinear model predictive control: closed-loop stability analysis”', *AIChE Journal*, 42(2), 3388-3402, (1996).

Smaili, A., and Mrad, F., Applied Mechatronics, MA: Oxford, (2008).

Soeterboek R., *Predictive Control: A Unified Approach*, U.S.A.: Prentice-Hall, Englewood Cliffs, (1992).

Tamimi, J. and Li, P., “A combined approach to nonlinear model predictive control of fast systems”', *Journal of Process Control*, 20(9), 1092-1102, (2010).

Venkataraman, P., *Applied Optimization with MATLAB Programming*, New York: Wiley-Interscience, (2002).

Vouzis, P.D., Bleris, L.G., Arnold, M.G., and Kothare, M.V., "A System-on-a-Chip Implementation for Embedded Real-Time Model Predictive Control", *IEEE Transactions on Control Systems Technology*, 17(5), 1006-1017, (2009).

XI, Y., LI, D. and LIN, S., "Model Predictive Control | Status and Challenges", *Acta Automatica Sinica*, 39(3), 222-236, (2013).

Yang, N., Li, D., Zhang, J. and Xi Y., “Model predictive controller design and implementation on fpga with application to motor servo system”, *Control Engineering Practice*, 20, 1229-1235, (2012).

Zeltom, “EMLS User Manual”, (19/03/2015), http://zeltom.com /documents/emls_um_14.pdf, Release 1.4, 2013, (2015).

9. ÖZGEÇMİŞ

Adı Soyadı : BEDRİ BAHTİYAR

Doğum Yeri ve Tarihi : ORDU – 10/10/1971

Lisans Üniversite : Marmara Üniversitesi

Y. Lisans Üniversite : Muğla Üniversitesi

Elektronik posta : bedribahtiyar@pau.edu.tr

İletişim Adresi : PAÜ Denizli TBMYO

Yayınlar:

Uluslararası ve ulusal hakemli dergilerde yaralan yayınlar:

- Bahtiyar, B., Çetin, A., Dombaycı, Ö.A., "Dokuma Tezgahlarından Denetleyici Alan Ağı ile Veri Toplama", *Pamukkale Üniversitesi Mühendislik Fakültesi Dergisi*, Cilt:13, Sayı:3, Sayfa:319-326, (2007).
- Bahtiyar, B., İplikçi, S., "An FPGA Implementation of Real-Time Nonlinear Runge-Kutta Model Predictive Control", *Transactions of the Institude of Measurement and Contro*l, (in press) (2015)

Uluslararası ve ulusal kongrelerde sunulan bildiriler:

- Bahtiyar, B., Iplikci, S., "An FPGA Implementation of Real-Time Nonlinear Runge-Kutta Model Predictive Control", *Control Automation Technologies (Contech'14),*Sayfa:41-53, 13 October, Istanbul, Turkey, (2014).
- Bahtiyar,B., İplikçi S., "Doğrusal Olmayan Runge-Kutta Model Öngörülü Kontrolün FPGA ile Gerçeklenmesi", *Türkiye Otomatik Kontrol Ulusal Toplantısı(TOK2014)*, 12 Eylül, Kocaeli, (2014).
- Çetin, A., Bahtiyar, B., "PV Sistemler için DAA Tabanlı İzleme Arayüzü Tasarım ve Uygulaması", *5. Uluslararası İleri Teknolojiler Sempozyumu (IATS'09),* 13-15 Mayıs, Karabük, Türkiye, (2009).
- Çetin, A., Bahtiyar, B., "Dokuma Salonu Ortam Bağıl Nem Oranının Denetleyici Alan Ağı ile Denetimi", *5. Uluslararası İleri Teknolojiler Sempozyumu (IATS'09)*, 13-15 Mayıs, Karabük, Türkiye , (2009).
- Çetin, A., Bahtiyar, B., Tenruh, M., "Fotovoltaik Sistem Verilerinin CAN Üzerinden Toplanması için Ağ Modelleri", *3. Ege Energy Symposium*, 25-28 May, Muğla/Türkiye, (2006).
- Bahtiyar, B., Çetin, A., "Tandem Tip Bir Tekstil Su Arıtma Tesisinin Mikro Denetleyici ve Bilgisayar ile Denetimi", *4rd International Advanced Technologies Symposium*, September 28-30, Konya / Türkiye , (2005).
- Büyüktuna, V., Yıldırım, O., Bahtiyar, B., "PIC Tabanlı Kepçe Tipi Rüzgar Hızı Ölçüm Cihazı Tasarımı", *I. Ege Enerji Sempozyumu ve Sergisi*, Mayıs 2003, Denizli , (2003).