Hareket Kontrol Sistemleri İçin Gerçek Zamanlı Haberleşme Yeteneğine Sahip Modüler Sistem Tasarımı

Modular System Design for Motion Control Systems with Real-Time Communication Capability

Muharrem Boyraz[1], *Denizhan Yerel*[2], *Ali Fuat Ergenç*[3],

[1]Gömülü Sistem Mühendisi
Repkon, Şile/İstanbul
muharrem.boyraz@repkon.com.tr

[2]Sistem Tasarım Mühendisi
Repkon, Şile/İstanbul
denizhan.yerel@repkon.com.tr

[3]Kontrol ve Otomasyon Müh.
Elektrik Elektronik Fak., İTÜ, Maslak/İstanbul
ali.ergenc@itu.edu.tr

Özetçe

Hareket kontrol sistemlerinin verimliliği ve güvenilirliği, çeşitli bileşenlerin sorunsuz entegrasyonu ve aralarındaki gerçek zamanlı iletişime dayanmaktadır. Bu çalışma, çeşitli alt sistemler arasında etkili iletişim ve koordinasyon sağlayan uygun maliyetli bir modüler hareket kontrol sistem tasarımı geliştirmeye odaklanmaktadır.

Modüler sistem tasarımı, dağıtılmış bir mimariyi içermekte olup sistemlerin ölçeklenebilir ve çeşitli uygulamalara adapte edilebilir olmasını sağlamaktadır. Her bir ünite, sensör verisi toplama, sinyal işleme veya sinyal üretme gibi işlevleri yerine getirmek üzere tasarlanmıştır. Ünitelerin birbirleri ile haberleşmeleri için görece uygun maliyetli ve tedarik sorunu yaşatmayacak komponentlerden oluşan Rbus (Repkon *Bus*) adında yeni bir haberleşme protokolü geliştirilmiştir.

Bu tasarımda kullanılan ve geliştirilen iletişim protokolleri, gerçek zamanlı gereksinimlere uygun olarak, modüller arasında belirlenebilir ve düşük gecikmeli iletişimi sağlamaktadır. Test amaçlı geliştirilen sistem, endüstriyel otomasyon uygulamalarında yaygın olarak kullanılan EtherCAT (*Ethernet for Control Automation Technology*) ve geliştirilen Rbus (Repkon *Bus*) protokollerinden yararlanmaktadır.

Abstract

The efficiency and reliability of motion control systems rely on seamless integration of various components and real-time communication between each other. This study focuses on developing a cost-effective modular motion control system design that enables effective communication and coordination among various subsystems.

Modular system design involves a distributed architecture, allowing the systems to be scalable and adaptable to various applications. Each unit is designed to perform functions such as sensor data acquisition, signal processing, or signal generation. A new communication protocol called Rbus (Repkon Bus) has been developed, consisting of relatively cost-effective and readily available components, to facilitate communication between the units.

The communication protocols used and developed in this design provide low-latency communication that can be determined between modules according to real-time requirements. The test system benefits from both EtherCAT (Ethernet for Control Automation Technology), widely used in industrial automation applications, and the developed Rbus (Repkon Bus) protocols.

1. Giriş

Hareket kontrol sistemi, bir veya daha fazla mekanik bileşeni kontrol etmek ve yönetmek için kullanılan bir sistemdir. Bu sistemler genellikle endüstriyel otomasyon, robotik, CNC (Bilgisayarlı Sayısal Kontrol) makineleri ve diğer benzer uygulamalarda kullanılır. Hareket kontrol sistemleri, önceden belirlenmiş bir hareketi gerçekleştirmek, konum veya hız gibi parametreleri kontrol etmek veya belirli bir hareket desenini takip etmek için kullanılmaktadır[1].

Hareket kontrol sistemi tasarımında, esneklik, ölçeklenebilirlik ve yüksek performans gibi faktörler büyük önem taşır. Geleneksel sabit yapılar, değişen gereksinimlere uyum sağlamada sınırlamaları olan bir tasarımı gerektirebilir[2]. Bu nedenle, modüler sistem tasarımı hareket kontrol sistemlerinin bu zorluklarını aşmak için kullanılan bir yaklaşımdır. Modüler

sistem tasarımı, hareket kontrol sisteminin bağımsız modüllerden oluşmasını ve bu modüllerin esnek bir şekilde bir araya getirilebilmesini sağlar.

EtherCAT protokolü günümüzde endüstriyel otomasyon uygulamalarında tercih edilen bir iletişim protokolüdür. Yüksek performanslı ve gerçek zamanlı veri alışverişi sağlama yeteneği, hareket kontrol sistemleri için önemli bir gereksinimdir[3].

Modüler sistem tasarımı, hareket kontrol sistemlerinde esnekliği sağlamanın yanı sıra, ölçeklenebilirliğini ve performansını artırarak endüstriyel otomasyon uygulamalarının daha verimli ve etkili bir şekilde çalışmasına olanak tanır. EtherCAT protokolü ise yüksek performanslı ve gerçek zamanlı veri iletişimi sağlayarak hareket kontrol sistemlerinin daha hassas ve hızlı tepki vermesini sağlar.

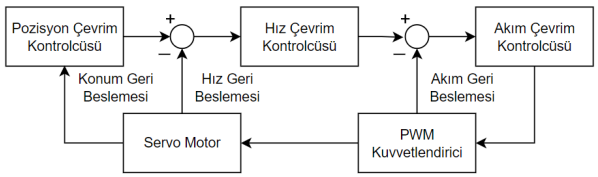
2. Modüler Sistem Tasarımı

Hareket kontrol sistemleri, sürekli olarak değişen gereksinimlere uyum sağlaması gereken sistemlerdir. Hareket kontrolü, bir sistemdeki nesnelerin veya mekanizmaların istenilen hareketleri gerçekleştirmek için kontrol edilmesini sağlayan bir süreçtir. Bu kontrol, pozisyon, hız ve ivme gibi parametrelerin belirlenmesi ve istenen değerlere ulaşmak için uygun komutların verilmesini içerir.

Hareket kontrolü genellikle motorlar, aktüatörler ve hareketli parçaların bulunduğu sistemlerde kullanılır. Endüstriyel robotlar, CNC makineleri, otomatik paketleme makineleri ve konveyör sistemleri gibi birçok uygulama, hareket kontrolünün kullanıldığı alanlardan sadece birkaçıdır.

Hareket kontrol sistemi, bir komut kaynağından gelen komutları alır, bu komutları işler ve ardından motorlar veya aktüatörler aracılığıyla mekanik hareketi gerçekleştirir. Hareket kontrolü, istenen hareketin hassas, doğru ve güvenilir bir şekilde gerçekleşmesini sağlamak için geri bildirim sensörlerini kullanabilir ve kontrol algoritmaları ile çalışabilir.

Hareket kontrolünün başlıca hedefleri arasında hızlı tepki süreleri, yüksek doğruluk, titreşimi azaltma, enerji tasarrufu ve güvenlik gibi faktörler bulunur. Bu amaçlar doğrultusunda, pozisyon, hız ve akım kontrol döngüleri gibi farklı katmanlarda kontrol gerçekleştirilebilir. Kullanılabilecek kontrol döngüleri Şekil 1'de ifade edilmektedir.



Şekil 1 Servo motor kontrol döngüleri

Konum kontrolü gerektiren uygulamalar için, konum hız döngüsünün etrafına bir konum/hız döngüsü eklenir. Konum döngüsü, gerçekleştirilen ve istenen konumlar arasındaki sapmayı belirler ve konum hatasını azaltmak veya ortadan kaldırmak için hız komutları üretir.

Hız döngüsü, en yaygın servo kontrol döngülerinden bir tanesidir. Komuta edilen hızı bir çözümleyici *resolver* veya kodlayıcı *encoder* aracılığıyla gerçek hızla karşılaştırır ve motorun hızını artırmak veya azaltmak için komutlar verir.

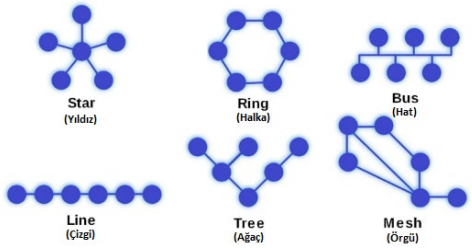
Akım kontrolü, birçok endüstriyel servo uygulamasında olduğu gibi tepki süresinin kısa olması gereken ve senkronizasyon gerektiren durumlarda kullanılmaktadır. Akım döngüsünün temel amacı torku kontrol etmektir, çünkü tork hızı etkiler ve dolayısıyla konum değişimi ortaya çıkar.

Herhangi bir kademeli sistemde, iç döngünün tepki süresi veya bant genişliği, dış döngünün tepki süresinden daha hızlı olmalıdır. Aksi takdirde, iç döngü dış döngü üzerinde çok az etkiye sahip olacaktır. İç içe geçmiş servo kontrol döngülerinde genel kural, hız döngüsünün konum döngüsünün 5 ila 10 katı kadar bir bant genişliğine sahip olması ve akım döngüsünün hız döngüsünün 5 ila 10 katı kadar bir bant genişliğine sahip olmasıdır[4].

Hareket kontrol sistemlerinde yüksek bant genişliği tercih edilmektedir. Ancak bir döngünün bant genişliği, içinde bulunan bir sonraki döngüyü etkilediği için, konum döngüsünün bant genişliğini artırmak, hız döngüsünün gereken bant genişliğini artırır. Benzer şekilde, hız döngüsünün bant genişliğini artırmak, akım döngüsü için gereken bant genişliğini artırır. Her iki durumda da, bir döngünün bant genişliğini bir sonraki, iç içe geçmiş döngünün gereken bant genişliğine ulaşılamayacak bir noktaya kadar artırmak, sistemin performansı için fayda sağlamaz. Haberleşme donanımlarının bant genişliğini olumsuz etkilememesi hedeflenir.

Modüler sistem tasarımı, hareket kontrol sistemlerinin esnekliğini ve ölçeklenebilirliğini artırmak amacıyla kullanılan etkili bir yaklaşımdır. Modüler sistem tasarımı, sistemin bağımsız modüllerden oluşmasını ve bu modüllerin kolaylıkla birleştirilebilmesini sağlar. Bu sayede, sistemde yapılacak güncellemeler daha hızlı ve düşük maliyetli bir şekilde gerçekleştirilebilir.

Şekil 2'de görülen yapıda haberleşme sistemlerinin destekleyebileceği bağlantı topolojileri görülmektedir. EtherCAT ağı, hızının yanı sıra, 65535 cihaza kadar bağlanabilir ve çizgi, hat, ağaç, yıldız veya bunların herhangi bir kombinasyonu bağlantı topolojisini desteklemektedir[5].



Şekil 2 Ağ topolojileri

Rbus ağının karmaşıklığı azaltmak ve geliştirme sürecini hızlandırmak amacıyla yalnızca çizgi (*line*) topolojisini desteklemesi planlanmaktadır.

Modüler sistem tasarımı için aşağıdaki adımlar izlenmektedir:

2.1. Modül Tasarımı

Bu çalışmada analog giriş-çıkış destekleyen motor sürücüleri kontrol edebilmek için ADC (*Analog-Digital Converter*) ve DAC (*Digital-Analog Converter*); motorların konum ve hız bilgilerini elde etmek için enkoder ve limit anahtarları gibi araçlar için dijital giriş-çıkış modülleri tasarlanmıştır. Bu şekilde referans hareket kontrol sistemlerinde bulunan öğeler hazırlanarak temel özellikler sağlanmaktadır.

Her bir modülün tasarımı hareket kontrol sisteminin esnekliğini sağlayabilecek ancak Rbus hattı üzerinde yedekli *redundancy* olmayacak şekilde tasarlanmıştır.

Modüllerin tasarımı için önce mikrodenetleyici modülleri tasarlanmıştır. Ana modüller üzerindeki mikrodenetleyiciler de tedarik sorunlarına karşı modüler yapıda tasarlanmasına karar verilmiştir. FPC (*Flexible Printed Circuit*) konektör bağlantı diziliminin belirlenmesi ve hem mikrodenetleyici hem de ana modüllerde bu yapı kullanılması planlanmıştır. Bu sayede bütün mikrodenetleyici modülleri ile ana modüllerin uyumlu olması sağlanmıştır. Mikrodenetleyici modüllerinin tasarımlarının ardından ADC, DAC, Enkoder, giriş-çıkış ve protokol dönüştürücü modüllerinin işlevlerini yerlerine getirmeleri için gerekli devreler tasarlanarak ve PCB üretimleri gerçekleştirilmiştir.

2.2. Haberleşme Protokolleri

Modüler sistem tasarımı sırasında uyulması gereken standartlar ve kullanılacak protokoller belirlenmiştir. Standartların takip edilmesi, modüllerin birbirleriyle uyumlu ve sorunsuz bir şekilde çalışması için büyük önem taşımaktadır. Endüstriyel otomasyon alanındaki mevcut standartlar ve protokoller göz önünde bulundurulmuştur. EtherCAT, PROFIBUS, SERCOS, CANOpen gibi birçok haberleşme protokolü bulunmaktadır. Ancak bu protokollerde uyumlu modül sayısı, tepki süresi, hata toleransı ve esneklik gibi parametreler göz önüne alındığında EtherCAT protokolü hareket kontrol sistemlerinde günümüzde kullanıma en uygun protokoldür. Hâlihazırda kullanılan protokollerin karşılaştırması Tablo 1'de listelenmiştir.

Protokol	Ağ İletim Hızı	Desteklenen Sistem Topolojileri	Bir Hatta Desteklenen Cihaz Sayısı	Ortalama Devir Süreleri
EtherCAT	100 Mbit/s *Full Duplex*	*Line, Ring, Tree, Star, Drop Line*	65535	76.6 µs - 300 µs
PROFINET RT	100 Mbit/s *Full Duplex*	*Line, Branch, Tree*	25	250 µs -512 ms
SERCOS III	100 Mbit/s *Full Duplex*	*Line, Ring*	511	1 ms - 65 ms
Can Bus	1 Mbit/s *Half Duplex*	*Line, Tree, Star, Multi-Master,*	127	1 ms - 10 ms

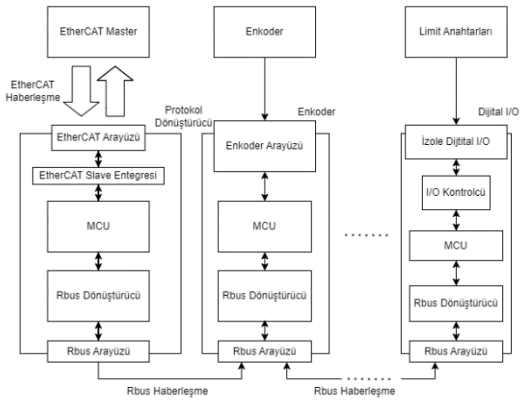
Tablo 1 Yaygın kullanılan protokollerin karşılaştırması

Gelecekte ortaya çıkabilecek gereksinimleri karşılamak ve oluşabilecek tedarik sorunlarına çözüm üretmek için EtherCAT protokolündeki gibi özelleşmiş entegre (*ASIC*) gerektirmeyen alternatif bir haberleşme hattının tasarlanmasına gerek olduğu görülmüştür.

Modüler sistem tasarımı başlığı altında anlatıldığı üzere tork kontrol çevrimi pozisyon ve hız kontrol çevrimine göre çok daha yüksek frekansta haberleşme gerektirmektedir. Bu nedenle tork kontrolüne ve yenilikçi kontrolcü yaklaşımlarının uyarlanmasına izin veren yüksek frekanslı bir haberleşme alt yapısı kurulmalıdır. EtherCAT protokolünün yüksek hızlı gerçek zamanlı haberleşmeyi desteklemesi sebebiyle Rbus protokolü ile EtherCAT birlikte kullanılmıştır. İki protokolün bir arada kullanımı senaryosu, Rbus üzerinden haberleşen modüller için bir protokol dönüştürücüsü tasarımını gerektirmiştir.

2.3. Fiziksel Bağlantı

Modüller arasındaki iletişimi sağlamak için uygun bir modül arabirimi tasarlanmıştır. Bu arabirim, veri ve kontrol sinyallerinin doğru şekilde iletilmesini sağlar. Arabirim, dijital veya analog sinyal; seri haberleşme veya paralel protokoller gibi farklı iletişim yöntemlerini içerebilir. Arabirimlerin kullanılacağı sistem şeması Şekil 3'te ifade edilmiştir.



Şekil 3 Rbus sistem şeması

Modüller arasında iletişimi sağlamak için arayüz olarak HDMI konektörü seçilebileceği tespit edilmiştir. HDMI konektör üzerinde 4 adet diferansiyel ve 11 adet tek uçlu iletim hattı bulunmaktadır. Tasarlanacak modüller arasında HDMI kablosu; protokol dönüştürücü ve endüstriyel bilgisayar arasında ise Ethernet kablosu ile bağlantı kurulmuştur. HDMI konektörüne alternatif konektörlerin karşılaştırması Tablo 2'de listelenmiştir.

Konektör	Çift Uçlu Hat Çifti Sayısı	Tek Uçlu Hat Sayısı	Boyut	Ortalama Konektör Maliyeti	Ortalama Kablo Maliyeti
HDMI	4	11	G: 14.50 mm Y: 04.55 mm D: 10.55 mm	~0,35 USD	~5 USD
USB Type-C	5	6	G: 08.94 mm Y: 02.54 mm D: 08.17mm	~2,98 USD	~20 USD
RJ-45	4	0	G: 15,15 mm Y: 11.65 mm D: 18.00 mm	~0,20 USD	~0,50 USD
DVI	7	15	G: 40.64 mm Y: 09.91 mm D: 18.40 mm	~1 USD	~5 USD

Tablo 2 Konektör karşılaştırmaları

3. Haberleşme Protokolü

Hareket kontrol sisteminde haberleşme protokolleri bölümünde anlatıldığı üzere EtherCAT protokolü ve geliştirilen Rbus protokolü birlikte kullanılmıştır. EtherCAT, Ethernet üzerinde çalışan, birçok alt protokolü sarmalayan ve gerçek zaman

desteği olan bir haberleşme protokolüdür. Bu haberleşme protokolü gerçek zamanlı ve gerçek zamanlı olmayan olmak üzere iki kısımdan oluşmaktadır.

Gerçek zamanlı olmayan kısım *mailbox* sistemi ile çalışmaktadır. *Mailbox*, parametre ve tanı verilerini iletmek için kullanılan bir EtherCAT alt sistemidir. CoE (*CAN over EtherCAT*), EoE (*Ethernet over EtherCAT*), FoE (*File over EtherCAT*) ve SoE (*SERCOS over EtherCAT*) alt protokollerini üzerinde bulundurmaktadır.

Gerçek zamanlı haberleşme için ise PDO (*Process Data Objects*) kullanılmaktadır. EtherCAT protokol dönüştürücü tasarımının yapılması için Rbus protokolünde kullanılacak olan EtherCAT alt unsurlarının CoE ve PDO olacağı ön görülmüştür.

3.1. Gerçek Zamanlı Olmayan Haberleşme

Tasarlanan sistemde gerçek zamanlı olmayan haberleşme için *mailbox* üzerinde alt protokollerden biri olan CoE kullanılabilmektedir. CoE, EtherCAT ağında CAN tabanlı cihazların kullanılmasını mümkün kılar ve bu sayede mevcut CAN sistemlerinin EtherCAT altyapısı üzerine taşınmasını sağlar. Bu iletişim profili, CAN protokolünün esneklik ve ölçeklendirilebilirlik gibi yaygın kabul gören özelliklerini EtherCAT ile birleştirir. Ortaya çıkan bu bileşim ile EtherCAT altyapısını kullanarak CAN tabanlı cihazlar arasında iletişim sağlanır.

3.2. Gerçek Zamanlı Haberleşme

PDO, EtherCAT ağı üzerindeki cihazlar arasında gerçek zamanlı veri alışverişini sağlar. Bu veri alışverişi, sorgu tekniği kullanılmadan *master* ile *slave* aygıtlar arasında verimli haberleşme gerçekleştirir. PDO, gerçek zamanlı verilerin iletimini hızlı ve güvenilir bir şekilde sağlar.

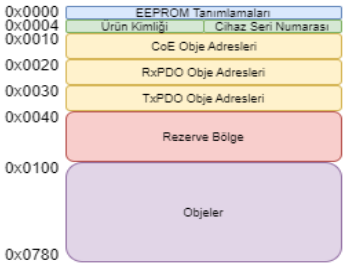
EtherCAT PDO, düşük gecikme süreleri ve yüksek hızlı veri iletimi sağlamak için optimize edilmiştir. PDO, veri alışverişini tek yönlü (örneğin, veri üreticiden tüketicilere doğru veya tüketiciden üreticiye doğru iletilir, aksi yönde veri aktarımı gerçekleşmez) veya çift yönlü olarak gerçekleştirebilir. PDO, EtherCAT ağı üzerindeki veri alışverişinde kullanılan veri yapısını tanımlamaktadır. PDO'lar, veri üreticisi ve tüketicisi arasında belirli veri tiplerini ve hedef adresleme mekanizmasını tanımlayan bir yapıya sahiptir. Bu sayede veri üreticisi, belirli bir PDO'ya veri yazabilir ve veri tüketicisi de aynı PDO'dan veriyi okuyabilir.[7]

EtherCAT PDO, endüstriyel otomasyon sistemlerinde gerçek zamanlı veri alışverişi sağlamak için yaygın olarak kullanılır. Yüksek performans, düşük gecikme süreleri ve esnek yapılarıyla EtherCAT PDO'ları, sistemler arasında hızlı ve güvenilir veri iletimini sağlar.

3.3. Protokol Dönüşümü

EtherCAT protokol dönüştürücü modülü, Rbus protokolü ile haberleşen hareket kontrol sisteminin EtherCAT ağına bağlanmasını ve veri iletişimini sağlamasını mümkün kılmıştır. EtherCAT protokolü, *master* cihaz (TwinCAT) ile haberleşmek için başlangıç anında *master* ve EtherCAT *slave* cihaz üzerinde bulunan veri formatının aynı olmasını gerektirir. Bu sebeple EtherCAT *slave* cihazlarının üzerinde haberleşmede kullanılacak veri formatının tutulduğu kalıcı hafıza bulunur. *Master* cihazın veri formatını ayarlaması için iki yöntem bulunmaktadır. Birisi statik olarak *slave* cihaz için daha önceden oluşturulmuş olan XML dosya türünde veri formatını içeren dosyayı kullanmaktır. Diğer yöntem ise dinamik olarak *slave* cihazdan veri formatını alarak, haberleşmeyi bu format ile başlatmaktır.

Rbus'ın modüler sistem yapısında eklenen her modül için veri yapısının güncellenmesi gerekmektedir. Bu gereksinim nedeniyle protokol dönüştürücü üzerinde statik veri formatını kullanmak mümkün değildir. Dinamik olarak veri formatını oluşturmak için EtherCAT *slave* olarak davranan protokol dönüştürücü modülün bütün Rbus modüllerinin veri yapısını bilmesi gerekmektedir. Bu nedenle bütün modüller üzerine bir adet kalıcı hafıza eklenmiştir. Kalıcı hafızada modülün veri yapısı protokol dönüştürücünün anlamlandırabileceği şekilde saklanmıştır. Kalıcı hafızada saklanan veri formatı, *mailbox* ve PDO haberleşmeleri sırasında alışverişi yapılacak olan verilerin uzunluğu, türü, indeksi ve değişkenlerin isimlerini tutmaktadır. Bu format oluşturulurken veriler EtherCAT protokolüne uygun bir şekilde oluşturulup kalıcı hafızalarda depolanmıştır. Kalıcı hafızanın içeriği Şekil 4'de ifade edilmektedir.



Şekil 4 Kalıcı hafıza içeriği

3.4. Haberleşmenin Başlatılması

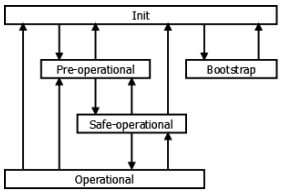
Protokol dönüştürücü EtherCAT haberleşmesini başlatmadan önce Rbus hattına bağlı modüller ile ilgili hazırlık yapmaktadır. İlk olarak modüller üzerinde bulunan kalıcı hafızalardan EtherCAT için gerekli veri formatı alınmaktadır. Alınan veriler EtherCAT haberleşmesi sırasında CoE ve PDO haberleşmelerinde kullanılacak değişkenlerin formatları ve sıralamasını tutmaktadır. Bu veriler EtherCAT formatına uygun şekilde yeniden düzenlenerek CoE ve PDO nesneleri oluşturulmaktadır. Ardından EtherCAT haberleşmesi başlatılmaktadır. EtherCAT haberleşmesi sırasında öncelikle CoE haberleşmesi başlatılarak CoE üzerinden PDO veri yapısı EtherCAT *master* cihaza iletilmektedir. Bu şekilde dinamik olarak veri yapısı oluşturulmaktadır. Dinamik yapı sayesinde ise Rbus hattına bağlanan modül sayısından ve çeşidinden bağımsız biçimde EtherCAT haberleşmesi başlatılabilmektedir. Dinamik olarak oluşturulan CoE verileri Şekil 5'te ifade edilmiştir.

1601:0	Encoder Outputs RxPDO-Map	RO	> 1 <
1601:01	SubIndex 001	RO	0x7010:01, 16
1602:0	DIO_PWM Outputs RxPDO-Map	RO	> 6 <
1602:01	SubIndex 001	RO	0x7011:01, 8
1602:02	SubIndex 002	RO	0x7011:02, 8
1602:03	SubIndex 003	RO	0x7011:03, 8
1602:04	SubIndex 004	RO	0x7011:04, 8
1602:05	SubIndex 005	RO	0x7011:05, 8
1602:06	SubIndex 006	RO	0x7011:06, 8
1603:0	ADC RxPDO-Map	RO	> 1 <
1603:01	SubIndex 001	RO	0x7012:01, 16
1604:0	DAC RxPDO-Map	RO	> 4 <
1A01:0	Encoder TxPDO-Map	RO	> 6 <
1A02:0	DIO_PWM TxPDO-Map	RO	> 3 <
1A03:0	ADC TxPDO-Map	RO	> 10 <
1A04:0	DAC TxPDO-Map	RO	> 2 <
1C00:0	Sync manager type	RO	> 4 <
1C12:0	RxPDO assign	RO	> 4 <
1C12:01	SubIndex 001	RO	0x1601 (5633)
1C12:02	SubIndex 002	RO	0x1602 (5634)
1C12:03	SubIndex 003	RO	0x1603 (5635)
1C12:04	SubIndex 004	RO	0x1604 (5636)

Şekil 5 TwinCAT 3 CoE verileri

3.5. EtherCAT Durum Makinesi

EtherCAT protokol dönüştürücü modül EtherCAT ile haberleşme sırasında EtherCAT durum makinesinin bulunduğu duruma göre farklı alt haberleşme protokollerini çalıştırabilir. Bu durumlar Şekil 6'da gösterilmiştir.



Şekil 6 EtherCAT durum makinesi[7]

EtherCAT protokolü *Init* durumunda ise *mailbox* ve PDO başlatılmaz. Eğer *Pre-OP* ise sadece *mailbox*, *Safe-OP* durumunda ise *mailbox* haberleşmesi ve PDO haberleşmesinin yalnızca giriş kısmı çalıştırılır. *OP* durumunda ise gerçek zamanlı ve gerçek zamanlı olmayan haberleşme çalıştırılır.

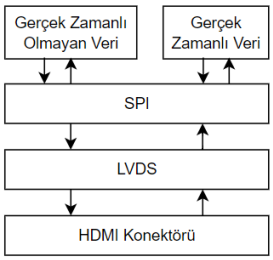
4. Rbus Protokolü

Rbus protokolü EtherCAT protokolünden gelen verileri modüllere dağıtabilecek şekilde tasarlanmıştır. Modüller arası haberleşmeyi sağlamak için çift yönlü *full-duplex* çalışabilen ve I2C, UART vb. protokollere göre çok daha yüksek hızlara çıkabilen tek uçlu yapıdan diferansiyel yapıya dönüştürülmüş SPI (*Serial Peripheral Interface*) kullanılmaktadır. Haberleşilecek modülün seçilmesi için ise kaydırmalı kaydedici ile oluşturulmuş modül seçme yapısı kullanılmaktadır.

Tek uçlu ile diferansiyel arasındaki dönüşüm için LVDS sinyali kullanılmaktadır. Protokol dönüştürücü modülde EtherCAT üzerinden alınan veri yeniden düzenlenerek SPI çevre birimi ile LVDS çevirici entegreye aktarılıp, LVDS ve kaydırmalı kaydedici sinyalleri HDMI konektör aracılığı ile bir sonraki modüle aktarılmaktadır.

EtherCAT üzerinden alınan veri yeniden formatlanırken hangi alt protokol ile alındığına bağlı olarak formatlama gerçekleştirilmektedir. Bunun sebebi ise Rbus protokolünde gerçek zamanlı olan ve olmayan alt protokollerin farklı şekilde ele alınması gereksinimidir. Protokollerin işletilebilmesi için modüller üzerinde EtherCAT durum makinesine benzer bir durum makinesi kurulmuştur. Kurulan durum makinesi gerçek zamanlı olan ve olmayan olmak üzere iki temel durumdan oluşmaktadır. Bütün modüller gerçek zamanlı olmayan durumda uyanmakta ve *master* tarafından EtherCAT CoE haberleşmesi ile parametre ayarlamalarına izin verebilmektedir.

EtherCAT durum kontrolcüsü OP moduna geçerken modüllere gerçek zamanlı haberleşme durumuna geçiş paketi göndermekte ve modüller kendi haberleşme paketlerini ve çevre birimlerini gerçek zamanlı haberleşme için yeniden ayarlamaktadır. OP modunda iken Rbus modülleri EtherCAT PDO haberleşmesinde olması gereken verileri protokol dönüştürücüye periyodik olarak göndermektedir. Böylece Rbus ile EtherCAT dönüşümü tamamlanmış olmaktadır. Rbus protokolünün temel aldığı fiziksel katman Şekil 7'de ifade edilmektedir.



Şekil 7 Rbus fiziksel katmanları

4.1. Parametrik-Gerçek Zamanlı Haberleşme Geçişi

Modüller üzerinde gerçek zamanlı ve gerçek zamanlı olmayan haberleşme durumlarıma göre çalışma sırasında SPI çevre birimi yeniden ayarlanmaktadır. Gerçek zamanlı olmayan haberleşme sırasında EtherCAT protokolü *slave* cihaza bir indeks için sorgu veya yazma isteği göndererek çalışmaktadır. İstek gönderilen indeksin gösterdiği verinin uzunluğu değişken olabileceği için gönderilip alınan paketlerin sabit bir uzunluğu bulunmamaktadır.

Gerçek zamanlı olmayan haberleşme sırasında modüller üzerindeki SPI çevre birimleri kesme modunda çalışmaktadır. Bu modda protokol dönüştürücü EtherCAT ile gelen veriyi haberleşme için anahtarlanan modüle gönderecek, modül kesme ile gelen veriyi yorumlayıp cevap için gerekli paketi oluşturacak ve gönderecektir.

Gerçek zamanlı haberleşmede paket uzunlukları haberleşme başlamadan belirlenmiş durumdadır. Bu sayede modül ile haberleşme sırasında alışverişi yapılacak veri paketi için sorgu veya yazma isteği gönderilen indekse/adrese göre değişiklik yapılmayacak, her veri alışverişi sırasında aynı uzunlukta paket gönderilecektir. Paket yapısının standart olmasının avantajı kullanılarak gerçek zamanlı haberleşme durumunda SPI çevre birimi DMA (*Direct Memory Access*) ile çalıştırılmıştır. DMA sayesinde SPI haberleşmesinin tepki süreleri düşürülmüş ve modül üzerinde bulunan mikrodenetleyici üzerindeki haberleşme yükü azaltılmıştır.

4.2. LVDS

LVDS (*Low Voltage Differential Signaling*), düşük gerilim farklılaştırma iletişim standardı olarak bilinen bir elektriksel arayüzdür. LVDS, veri iletimi için yüksek hızlı, düşük güç tüketimine sahip ve gürültüye karşı dirençli bir yöntem sağlar.

LVDS, diferansiyel sinyal çiftlerini kullanarak veri iletimini gerçekleştirir. Bu çiftlerde her bir sinyal hattı için bir pozitif ve bir negatif hat bulunur. Sinyal iletimi, iki hattın gerilim farkının değerine bağlı olarak gerçekleşir. LVDS, pozitif ve negatif hat

arasındaki gerilim farkını algılayarak veriyi iletim hattına kodlar ve alıcı tarafında bu gerilim farkını yorumlayarak gönderilen ham veriyi elde eder.

LVDS, veri iletim hızları 155 Mbps ile 3,125 Gbps arasında değişebilen yüksek hızlı haberleşmeye imkân tanır. Düşük güç tüketimi sağlar ve yüksek veri bant genişliğiyle birlikte gürültüye karşı dayanıklıdır.

4.2. Modül Seçim Yapısı

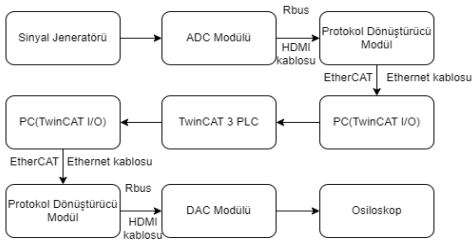
Rbus hattına bağlanan cihazlar için haberleşme SPI çevre birimi temel alınarak gerçekleştirilmektedir. SPI, haberleşmeyi başlatmak için saat ve veri sinyallerine ek olarak CS (*Chip Select*) adı verilen hat *bus* topolojisinde hangi *slave* cihaz ile haberleşileceğinin seçilmesi için kullanılan bir anahtarlama sinyalini gerektirmektedir. Bu sinyal her *slave* için özel olmak zorunda ve standart kullanımda her cihaz için *master* cihazdan bir fiziksel çıkış bağlantısı gerektirmektir. Modüler sistemde bu durum SPI protokolünün kullanımını zorlaştırmaktadır ve modüller arasında bağlantı sayısı sınırlandırmaktadır. Rbus sisteminde CS için her modül üzerinde bir adet kaydırmalı kaydedici bulunmaktadır. Kaydırmalı kaydedici sayesinde protokol dönüştürücü modülde sadece 3 tekil hat kullanılarak haberleşme protokolünün gerektirdiği CS sinyali hatta bağlı modüller üzerinde oluşturulabilmektedir.

SPI CS sinyaline bağlı cihaz sayısı sınırı problemi *master* cihazın HDMI konektöründe bulunan tekil hat sayısından bağımsızlaştırılmıştır. Alınan tasarım kararı sayesinde, bütün modüller arasında diferansiyel SPI ve kaydırmalı kaydedici kontrolü için gerekli sinyaller aktarılabilecektir.

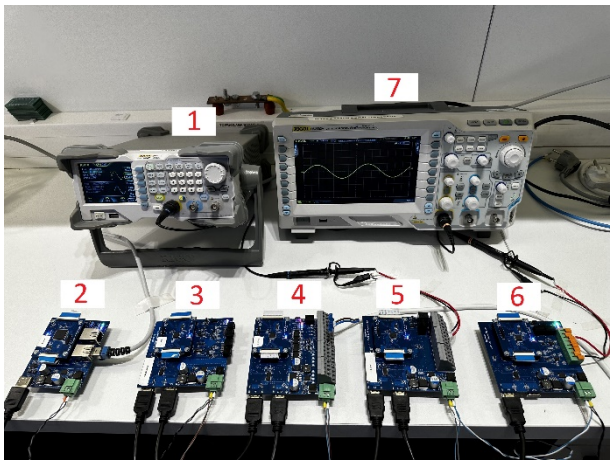
5. Test

Tasarımı yapılan modüllerin, modüler sistemin isterlerini sınamak amacıyla farklı kombinasyonlar oluşturularak haberleşme testleri gerçekleştirilmiş ve başarılı sonuçlar alınmıştır. Modüller arasında yapılan haberleşme testlerinin ardından protokol dönüştürücü modülü test edilmiştir. Kurulan test düzeneğinde EtherCAT protokolü Rbus protokolüne dönüştürülerek haberleşme sağlanmıştır.

Haberleşme testi sırasında masaüstü sinyal jeneratörü kullanılarak üretilen sinyal, ADC modülü ile örneklenerek Rbus protokolü ile protokol dönüştürücü modüle gönderilmiştir. Rbus protokolündeki veriler protokol dönüştürücü kartta EtherCAT protokolüne dönüştürülmüş ve EtherCAT ile TwinCAT 3 programını çalıştıran bilgisayara gönderilmiştir. Test bilgisayarına gelen veri TwinCAT 3 PLC ile Rbus DAC modülüne geri yönlendirilmiştir. Kurulan test düzeneğinin şeması Şekil 8'de ve görseli Şekil 9'da ifade edilmektedir.



Şekil 8 Test akış şeması



Şekil 9 Rbus haberleşme test sistemi. 1. Sinyal jeneratörü, 2. Rbus protokol dönüştürücü, 3. Enkoder okuyucu, 4. dijital giriş-çıkış modülü, 5. ADC modülü, 6. DAC modülü, 7. Osiloskop

6. Sonuç

Hareket kontrol sistemleri için gerçek zamanlı haberleşme yeteneğine sahip modüler sistem tasarımı projesinin sonucunda bir adet protokol dönüştürücü, ADC, DAC, enkoder okuyucu ve dijital giriş-çıkış modülü tasarımı yapılmıştır.

Teşekkür

Sakarya Üniversitesi Elektrik & Elektronik Mühendisliği öğrencisi Salih Eren Sağırlı'ya emeklerinden dolayı teşekkürlerimizi sunarız.

Kaynakça

[1] Motion Control Handbook, William Y. Svrcek, Joseph L. Staud, Michael McCarthy, ve C. W. de Silva, 2007.

[2] https://ieeexplore.ieee.org/abstract/document/7301614 "Scalable Motion Control System Using EtherCAT-based Shared Variables"

[3] https://digital-library.theiet.org/content/journals/10.1049/cce_20040104 "Real-time Ethernet: the EtherCAT solution"

[4] https://www.motioncontroltips.com/faq-servo-motor-current-velocity-position-loops-bandwidths/ "What are servo motor current, velocity and position loops and bandwidths?"

[5] https://www.beckhoff.com/media/downloads/information-media/etg_brochure_en.pdf, s:3

[6] https://infosys.beckhoff.com/english.php?content=../content/1033/bk51x0/2519207947.html&id=

[7] https://infosys.beckhoff.com/english.php?content=../content/1033/ax5000_usermanual/html/Bt_EcBasics_EcStateMachine.htm&id

[8] https://tr.wikipedia.org/wiki/Serial_Peripheral_Interface

[9] https://studymuch.in/network-topology/