YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PARALEL KANAL - PCI YEREL YOLU BAĞLANTISININ GERÇEKLENMESİ İLE ANA BİLGİSAYAR SİSTEMİNİN PCI UYUMLU ÇEVRE BİRİMLERİNİ KULLANMASINA YÖNELİK BİR SİSTEM

Bilgisayar Yük.Müh. Ali Gökhan YAVUZ

F.B.E. Bilgisayar Bilimleri Mühendisliği Anabilim Dalında
Hazırlanan

DOKTORA TEZİ

Tez Savunma Tarihi	: 17 Aralık 1999
Tez Danışmanı	: Prof. Mehmet Yahya KARSLIGİL (Y.T.Ü.)
Jüri Üyeleri	: Prof. Dr. Oğuz TOSUN (B.Ü.)
	Doç. Dr. Bülent Örencik (İ.T.Ü.)

İSTANBUL, 1999

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

TEŞEKKÜR

Yıldız Üniversitesi Bilgisayar Bilimleri ve Mühendisliği Bölümü'ne girdiğim günden bugüne kadar, bilgisi, değerli görüşleri ve tecrübesi ile beni her konuda yönlendiren, öğrencisi ve asistanı olma onurunu taşıdığım hocam Prof. M. Yahya KARSLIGİL'e beni bilimsel açıdan doktora yapabilecek bilgi ve beceri seviyesine getirdiği ve doktora tezimin oluşmasında ve hazırlanmasında değerli yardımları ile yol gösterdiği için çok teşekkür ederim.

Son bir yıldır, neredeyse her gün tezimi kaleme almam konusunda beni uyararak bu kitabın oluşmasını sağlayan, hem tezimle ilgili yaptığım araştırma ve çalışmalarda hem de tez kitabımın hazırlanmasında en az benim kadar ilgi ve titizlikle bana yardım ve desteğini sunan sevgili arkadaşım, Y.T.Ü. Bilgisayar Bilimleri ve Mühendisliği Bölümü öğretim üyesi Yrd. Doç. Dr. M. Elif KARSLIGİL'e çok teşekkür ederim.

Haftanın 6 günü, sabahlara kadar üniversitede çalıştığım için ancak pazar günleri görüşebildiğin sevgili anneme ve babama, bana doğruluk, dürüstlük ve çalışkanlık ile her zorluğun üstesinden gelinip her kapının açılabileceğini öğrettikleri ve iyi bir öğrenim alabilmem için hiç bir özveriden kaçınmadıkları için, onları hayal kırıklığına uğratmamış olmak dileği ile, çok teşekkür ederim.

ÖZET

Bu çalışmada, IBM S/390 ana bilgisayar sistemlerinin paralel kanal giriş_çıkış altyapısının, kişisel bilgisayar sistemlerindeki PCI yerel yolu arasındaki fiziksel bağlantının gerçeklenmesi ve sistemler arası protokol farklılıklarının giderilmesi ile ana bilgisayar sistemlerinin, kişisel bilgisayar çevre birimlerini kullanması için yeni bir sistem geliştirilmiş ve sistemin tasarımında ve gerçeklenmesinde elde edilen sonuçlar değerlendirilmiştir.

Geliştirilen sistemin tasarımında, veri haberleşmesinde farklı topolojiye ve protokole sahip yapılar arasında geçişin sağlanmasına benzer bir yaklaşımla kolay tasarlanabilir, gerçekleştirilebilir, genişletilebilir ve test edilebilir olması amacıyla katman modelinden yararlanılmıştır. Sistemde olması istenen özellikler ve fonksiyonlar katman yapısına göre değerlendirilip sınıflandırıldığında **fiziksel katman**, **protokol katmanı**, **dönüşüm katmanı** ve **uygulama katmanı** olarak isimlendirilen 4 katmana ihtiyaç olduğu belirlenmiştir.

Fiziksel katman ile, paralel kanal ve PCI yerel yolu arasındaki elektriksel bağlantı sağlanmıştır. Protokol katmanında, fiziksel katmandan gelen sinyaller değerlendirilmiş ve anlamlı hale getirilmiştir. Dönüşüm katmanında kişisel bilgisayar sistemlerindeki çevre birimleri ile ana bilgisayar sistemlerindeki çevre birimleri arasındaki yapısal ve fonksiyonel farklılıklar ortadan kaldırılmıştır. Uygulama katmanı ise ana bilgisayar giriş_çıkış isteklerini ilgili kişisel bilgisayar birimlerinde başlatmak, yürütmek ve sonlandırmak amacı ile oluşturulmuştur.

Katmanlar, bu tez çalışmasında geliştirilen özel donanım ve yazılımlar ile gerçeklenmiştir. Paralel kanal ile kanal protokolüne uygun olarak fiziksel bağlantıyı kurmak, veri aktarımını gerçekleştirmek ve benzeri işlemleri yürütmek amacıyla, programlanabilir donanımdan da yararlanılarak "Parallel Channel Interface Processor" (**PCIP**) olarak isimlendirilen özel bir birim tasarlanmış ve gerçeklenmiştir.

Katmanlar için geliştirilen yazılım modüllerinin çalıştırılması, aralarında senkronizasyonun sağlanması, donanım ve yazılım kaynaklarına erişimin kontrol edilmesi ve bu kaynakların paylaştırılması amacı ile gerçek-zamanlı çalışmaya uygun, çoklu programlama (multiprogramming) ortamını destekleyen, "Service Processor Control Program" (**SPCP**) adı verilen özel bir kontrol programı gerçekleştirilmiştir.

ABSTRACT

In this thesis, a new system for the interconnection of the input / output subsystem of IBM S/390 mainframe systems, with the PCI local bus system of personal computers has been proposed. The proposed system enables mainframe systems to access PCI compliant I/O devices as they were directly attached to mainframes parallel channel I/O subsystem.

For the design of the proposed system, a layered approach has been used, where the layer structure constructed is in analogy with layer structure found in interconnection of different topologies and protocols common to data communication applications. The layered approach facilitated the design, implementation, expansion and verification of the whole system. A classification of the functions that constitute the system revealed that 4 layers are needed. Namely, physical layer, protocol layer, conversion layer and application layer.

Physical layer is developed to match the electrical characteristics of the parallel channel and PCI local bus. At protocol layer, the electrical signals received from the physical layer are converted into meaningful data. Conversion layer is built to accomplish the transparent conversion from mainframe peripherals to the PCI compliant peripherals, where their fucntional differences are balanced. Application layer is developed to handle I/O requests by supervising the initiation, execution and termination of them on PCI compliant peripherals.

Special hardware and software, which has been developed in this thesis, has been used in the realization of the layers.

A specific control unit, which is named "Parallel Channel Interface Processor" (**PCIP**) has been designed and implemented via programmable hardware to establish the physical link with the parallel channel, to transfer data to/from parallel channel and to accomplish similar tasks related to the parallel channel.

"Service Processor Control Program" (**SPCP**), which supports a multiprogramming environment for real-time tasks, was also developed to supervise and synchronize the execution and interaction of software modules, responsible of the functioning of the layers, by ensuring correct resource sharing.

1. GİRİŞ

2000 yılı ile yeni bir 1000 yılın kapılarını aralamaya hazırlanırken geride bıraktığımız son 10 yıl içerisinde bilgisayar ile ilgili donanım ve yazılım teknolojilerinde elde edilen hızlı ilerlemeler, bilgisayar sistemlerinde çok yönlü değişim ve gelişmeler olmasını sağlamıştır.

Günümüzden 10 yıl öncesinde, işlem hızı, bellek ve veri depolama kapasitesi, güvenirlik ve birden fazla kullanıcıya hizmet vermenin ön planda olduğu bilgisayar uygulamalarında, sadece ana bilgisayar sistemleri kullanılabilmekteydi. Ana bilgisayar sistemlerinin, havalandırmalı, yükseltilmiş tabanlı özel mekanlarda çalıştırılmak zorunda olmaları, bakım, yenilenme, işletme ve diğer maliyetlerinin yüksek olması bu sistemlerin kişisel kullanımdan uzak, özel uygulamalar için kurulmalarını gerektirmiştir. Bilginin ve buna bağlı olarak bilgi teknolojilerinin kurumların organizasyonunda giderek önem kazanması ve vazgeçilmez olmaya başlaması, yüksek maliyet gerektiren ana bilgisayar sistemlerine yatırım yapamayan kurumlar için alternatif bilgisayar sistemlerinin geliştirilmesini zorunlu kılmıştır.

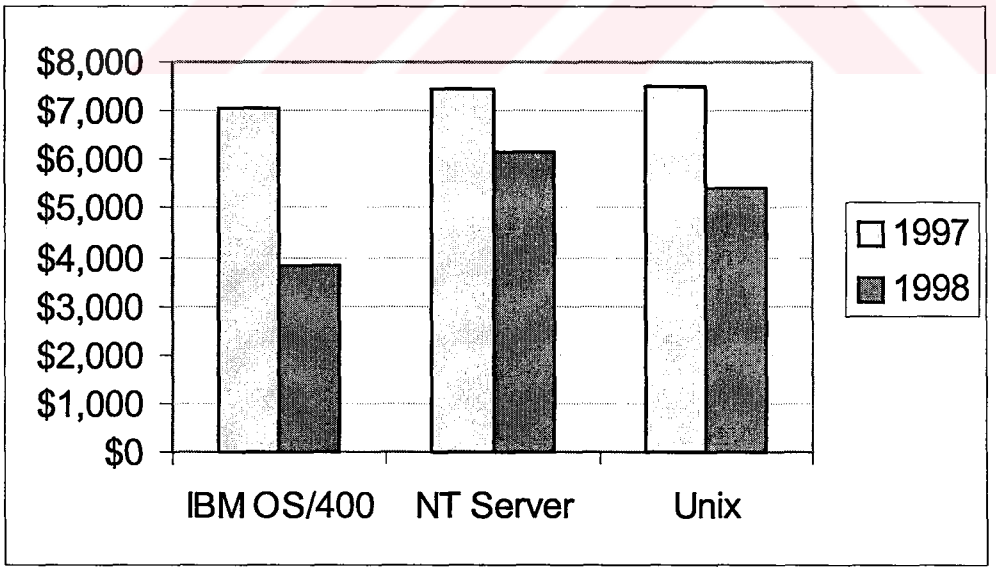
Özel mekan gerektirmeyen, herkes tarafından kolaylıkla kullanılabilen, bakım ve işletmesi düşük maliyetli kişisel bilgisayar sistemleri, gelişen teknolojiye paralel olarak işlemcilerinin, bellek ve çevre birimlerinin hızları, kapasiteleri ve güvenirlikleri arttırılarak, zaman içerisinde ana bilgisayar sistemlerine yakın performans ve güvenirliğe getirilmişlerdir. Donanım alanında elde edilen bu gelişmelerle beraber kişisel bilgisayar sistemleri için geliştirilen işletim sistemleri de, bir taraftan, ana bilgisayar sistemlerinde olduğu gibi, aynı anda birden fazla kullanıcıyı ve işlemi desteklerken diğer taraftan kullanımı daha kolay hale getirmek için grafik arayüzüne dayalı bir çalışma ortamı oluşturmuşlardır.

Kişisel bilgisayar sistemlerinde yazılım ve donanımda elde edilen bu gelişmeler, aynı dönemde geniş ve yerel alan ağlarında haberleşme konusundaki ilerlemeler ile desteklenmiştir. Bilgisayar ağlarının haberleşme hızları artarken, bilgiye konumlarından bağımsız olarak erişmeyi, bu bilgileri değişik kullanıcılar arasında paylaştırmayı imkanlı

kılan haberleşme protokolleri de geliştirilmiş ve yaygınlaşmıştır. Bu gelişim içerisinde kişisel bilgisayar sistemleri sahip oldukları işlem gücü sebebiyle o zamana kadar alışılagelmiş tek merkezden kontrol edilen yapıların aksine, işlemlerin birbirlerine eşdeğer sistemlere dağıtılması prensibine dayalı istemci_sunucu (client / server) ve dağıtılmış işleme (distributed processing) modellerinin oluşturulmasında önemli rol oynamışlardır.

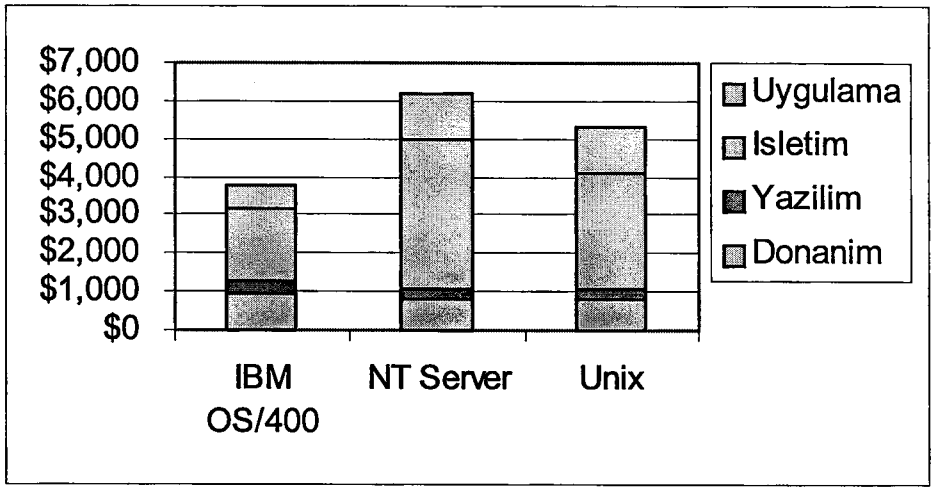
Ana bilgisayar sistemleri ile istemci_sunucu modeline göre kurulmuş kişisel bilgisayar tabanlı sistemlerin karşılaştırılması, istemci_sunucu modeline geçilmesinin kurumlar açısından daha yararlı ve düşük maliyetli olacağını ortaya koymuşsa da ilerleyen yıllar içerisinde bu analizin kısa döneme yönelik bir analiz olduğu, uzun dönemde kişisel bilgisayar sistemlerine dayalı istemci_sunucu modelinin işletim, bakım ve kullanım yönünden daha fazla kaynak gerektirdiği ortaya çıkmıştır.

International Data Corporation (IDC) tarafından 1998 yılında yapılan bir araştırmada (IDC, 1998), ana bilgisayar sunucu sistemlerinin genel maliyetlerinin, 1997-1998 yılları arasında yaklaşık %50 azalma gösterirken, WindowsNT veya Unix sunucu sistemlerinde bu azalmanın sırasıyla %17 ve %28 oranlarında kaldığı belirlenmiştir. (Şekil 1.1)



Şekil 1.1 Değişik sunucu platformlarında, 1997-1998 yıllarında gözlenen maliyet değişimi (IDC, 1998)

Toplam sunucu maliyetlerinde, donanım, yazılım, işletme ve uygulama geliştirme giderlerinin sunucu türlerine göre dağılımı ise Şekil 1.2'de görülmektedir.



Şekil 1.2 Değişik sunucu platformlarında donanım, yazılım, işletme ve uygulama geliştirme giderlerinin dağılımı (IDC, 1998)

İstemci_sunucu modelinin düzgün işleyebilmesi ve yüksek performans sağlayabilmesi istemcilerden çok sunucunun görevi olduğu için, özellikle son yıllarda ana bilgisayar sistemleri bu model içerisine sunucu görevini yerine getirecek şekilde entegre edilmeye başlanmışlardır. Bu şekilde oluşturulan yeni model, ağ merkezli (network centric) model olarak da isimlendirilmektedir. Ağ merkezli model, "Web" sitesi yönetimi, veri tabanı yönetimi, ağ yönetimi, veri güvenliği ve sunucu sisteminin genişletilebilme özellikleri öncelikli olmak üzere sunucu sistemlerinde pek çok değişik özelliğin bir arada olmasını gerektirmektedir. Değişik sunucu platformlarında işletme giderlerinin dağılımına ait IDC araştırmasının (IDC, 1998) sonucu **Tablo 1.1**'de verilmiştir.

Tablo 1.1 Değişik sunucu platformlarında işletme giderlerinin dağılımı (IDC, 1998)
(Ağ merkezli modelde önemli olan unsurlara ait giderler açık gri ile belirtilmiştir.)

Maliyet Unsurları	IBM OS/400		NT Server		Unix	
	$/Yıl/Kişi	%	$/Yıl/Kişi	%	$/Yıl/Kişi	%
Planlama ve Yönetim	416	17	831	17	621	15
Satın alma	26	1	248	5	110	3
Kuruluş	162	7	312	6	293	7
Kullanıcı Destek	424	17	754	15	532	13
Uygulama Geliştirme	723	29	1157	23	1321	32
Web Yönetimi	160	6	171	3	132	3
Veri Tabanı Yönetimi	123	5	394	8	247	6
Yönetim	98	4	236	5	217	5
Ağ Yönetimi	125	5	319	6	213	5
Yedekleme	25	1	40	1	80	2
Fon Yönetimi	0	0	79	2	88	2
Geliştirme/ /Değiştirme	32	1	146	3	91	2
Operasyon	148	6	315	6	169	4
Devre Dışı Bırakma	49	2	32	1	25	1
İmha	0	0	0	0	34	1
TOPLAM	2511	100	5034	100	4183	100

Hem istemci_sunucu modeline hem de ağ merkezli modele göre sunucu sistemlerinin güvenilirliği ve plansız devre dışı kalma süresinin kısalığı bu modellerin efektif çalışmasında önemli rol oynamaktadır. Gartner (1998) tarafından yapılan araştırma sonuçlarına (Tablo 1.2) göre ana bilgisayar sistemlerinde yıl içerisinde plansız devre dışı kalma süresi dakika ile ifade edilirken, kişisel bilgisayar sistemlerinde bu süre gün ile ifade edilmektedir.

Tablo 1.2 Değişik Sunucu Platformlarının Güvenilirlikleri (Gartner, 1998)

Platform	Plan Dışı Durma/Sunucu/Yıl	Güvenilirlik (24 saat x 365 gün)
IBM S/390 (sysplexed)	0.17 Saat	%99.998
Tandem	1.7 Saat	%99.98
IBM AS/400	5.2 Saat	%99.94
IBM S/390 (non-sysplexed)	8.9 Saat	%99.90
VAX	18.9 Saat	%99.78
UNIX	23.6 Saat	%99.73
NT Sunucu	224.5 Saat	%97.44

Gerçekleştirilen bu doktora tezinin konusu "Paralel Kanal - PCI Yerel Yolu Bağlantısının Gerçeklenmesi ile Ana Bilgisayar Sisteminin PCI Uyumlu Çevre Birimlerini Kullanmasına Yönelik Bir Sistem" dir. Geliştirilen bu sistem, ana bilgisayar sistemlerinin, kişisel bilgisayar sistemlerinde yer alan donanım kaynaklarını yazılıma saydam kullanabilmelerini ve bu kaynakları kişisel bilgisayarda çalışan uygulamalar ile paylaşabilmelerini sağlamaktadır. Gerçekleştirilen sistemde, ana bilgisayar sistemi giriş_çıkış altyapısı ile PCI (Peripheral Component Interconnect) yerel yolu arasında, topolojiden ve protokolden kaynaklı farklılıklar giderilerek fiziksel bağlantı kurulmuş ve PCI uyumlu kişisel bilgisayar çevre birimlerinin, ana bilgisayar sistemine uyumlu olmaları sağlanmıştır. Tasarlanan sistem, bir giriş_çıkış biriminin görevlerinin başka bir giriş_çıkış birimi tarafından gerçekleştirilebilmesi durumunda, ana bilgisayar sisteminden gelen istekleri, sistem konfigürasyonuna bağlı olarak bu iki birimden birine saydam olarak yönlendirebilmektedir. Bu yönlendirmede ihtiyaç duyulan dinamik dönüşüm geliştirilen sistem tarafından yapılmaktadır. Bu sayede, ana bilgisayar sistemi bir teyp birimine eriştiğinde, bu erişim isteği PCI tarafında bir teyp biriminde gerçekleştirilebileceği gibi, bir disk birimine de yönlendirilebilmektedir.

Kaynakların dinamik atanması ve bir kaynak yerine uyumlu bir diğer kaynağın kullandırılabilmesi sayesinde gerçekleştirilen sistem, kaynaklarda hata oluşması durumunda da çalışmasını sürdürebilmektedir. Bu özellik, sisteme hatalara toleranslı (fault tolerance) olma imkanı sağladığı gibi verilerin tek bir yapı altında kolayca erişilebilir şekilde saklanmasını (dataware housing) gerektiren uygulamalar için de kolaylık sağlamaktadır. Birden çok, benzer veya farklı yapıda giriş_çıkış biriminin, aynı anda desteklenmesi ve ana bilgisayar sistemine kullandırılması da gerçekleştirilen sistemin önemli özelliklerindendir.

Bu konu ile ilgili yapılmış olan çalışmalar incelediğinde, MicroChannel mimarisine uygun kişisel bilgisayarlar için gerçekleştirildikleri, teyp birimi veya terminal kontrol birimi olarak çalışabildikleri ancak birden fazla farklı tipte giriş_çıkış birimini desteklemedikleri belirlenmiştir. MicroChannel mimarisi, günümüzün yüksek hızlı kişisel bilgisayar giriş_çıkış birimlerinin veri aktarım hızlarına uyum sağlayamamasının yanı sıra teknolojik ömrünü doldurmuş ve standart olarak kabul görmemiştir. Kişisel bilgisayar sistemlerine ait

çevre birimlerinde son yıllardaki gelişmeler, bu birimlerin hem kapasitelerini hem de performanslarını ana bilgisayar sistemlerininkine eşdeğer bir seviyeye taşıdığı için, sadece teyp birimlerinin veya terminal birimlerinin ana bilgisayar sistemine kullandırılması yeterli olmayacaktır.

Bu tez çalışmasında önerilen ve gerçekleştirilen sistem, kişisel bilgisayar alanında endüstri standartı olan, yüksek hızlı veri aktarımı ve giriş_çıkış işlerinin işlemciden bağımsız yapılabilmesi özelliklerine sahip PCI yerel yolunun kullanılması, ana bilgisayar sistemine tanıtılacak PCI çevre birimlerinin tipinde bir kısıtlama getirilmemesi, aynı anda birden fazla farklı tipte çevre biriminin desteklenmesi, dinamik olarak kaynaklar arasında atama yapılabilmesi, hata durumlarına karşı önlem alınabilmesi ve "dataware housing" uygulamalarına uygun tasarlanmış olmasıyla mevcut sistemlerdeki açıkları kapatmasının yanısıra yenilik ve üstünlükler de getirmektedir.

Gerçekleştirilen sistemin tasarlanmasında, öncelikle problemin tanımı ortaya konmuş ve çözümü için bir model oluşturulması yoluna gidilmiştir. Paralel kanal ve PCI yerel yolu mimarilerinin incelenmesi sonunda, problemin veri haberleşmesindeki farklı topoloji ve protokole sahip yapıların birleştirilmesine benzer şekilde ele alınabileceğine ve çözüm için katman yapısı uygulanmasına karar verilmiştir. Fiziksel katman, protokol katmanı, dönüşüm katmanı ve uygulama katmanı olmak üzere 4 farklı katman oluşturulmuştur. Bu katmanların gerçeklenmesinde donanım ve yazılımlardan yararlanılmıştır. Ana bilgisayar sistemlerindeki paralel kanal yapısına bağlantıyı sağlamak amacıyla, "Paralel Channel Interface Processor" **(PCIP)** olarak isimlendirilen özel bir donanım biriminin tasarımı da gerçeklenmiştir.

Gerçeklenen donanım birimlerinin tasarımlarının algoritmik yapılarla ifade edilerek yazılım ile çözülmüş ve programlanabilir donanım ile uygulanmış olması da bu tez çalışmasının önemli bir özelliğidir. Bu sayede problemler, "böl ve yönet" mantığına uygun olarak modüler parçalar halinde çözülmüş ve ulaşılan çözümlerin doğruluğu yazılımla test edilmiştir.

Katmanlara ait yazılım modüllerinin çalıştırılması, bu modüller arasında senkronizasyonun sağlanması, donanım ve yazılım kaynaklarına erişimin kontrol edilmesi ve bu kaynakların paylaştırılması amacı ile oluşturulan modelde tanımlanan servis işlemcisinde çalıştırılmak üzere, gerçek-zamanlı çalışmaya uygun, çoklu programlama (multiprogramming) ortamını destekleyen, "Service Processor Control Program" (**SPCP**) adı verilen özel bir kontrol programı gerçekleştirilmiştir.

Bu tez çalışması, PCI yerel yolu ve ana bilgisayar giriş_çıkış sistemine bağlı bir çalışma olduğundan öncelikle bu yapılar sırasıyla 2. ve 3. bölümde tanıtılmışlardır. 4. bölümde sistemin modellenmesi yapılmıştır. 5. bölümde gerçekleştirilen sistemin donanım özellikleri, 6. bölümde ise yazılım özellikleri incelenmiştir. 7. bölümde bu tez çalışmasında elde edilen sonuçlar değerlendirilmiştir.

2. PCI YEREL YOLU

Bu tez çalışmasında geliştirilen sistemin değerlendirilebilmesi için PCI (Peripheral Component Interconnect) yerel yolu ile ilgili temel kavramların bilinmesi gereklidir. Bu bölümde PCI yerel yolunun yapısı ile özellikleri ve bu yolu oluşturan sinyaller, komutlar incelenecek, PCI birimlerinin adreslenmesine ve veri aktarımlarına yer verilecektir.

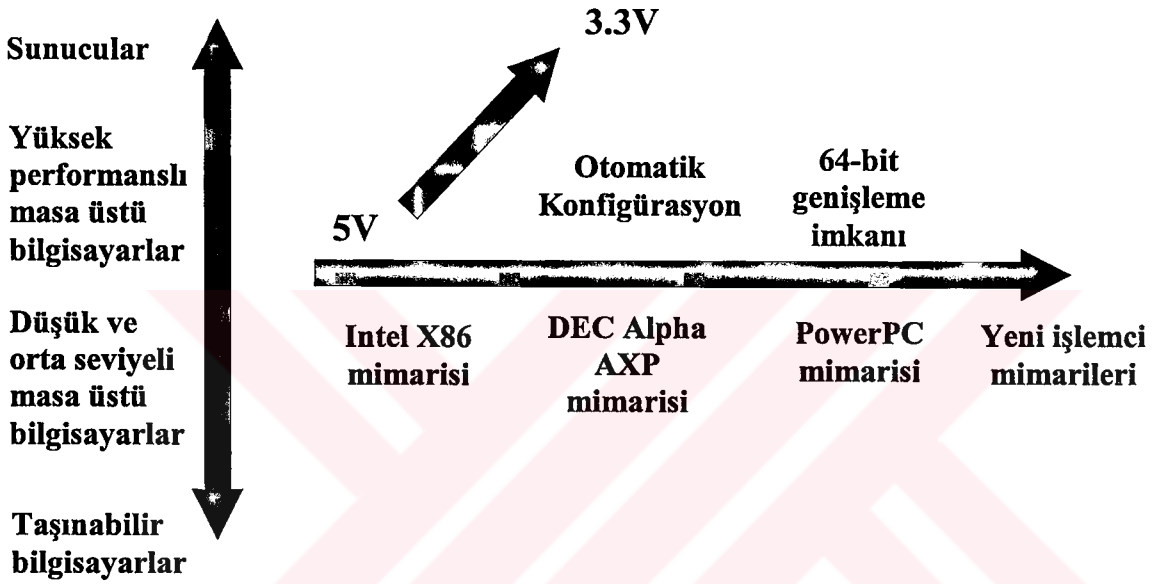
2.1 PCI Yerel Yolunun Yapısı ve Özellikleri

Son yıllarda özellikle grafik arayüzüne sahip işletim sistemlerinin ve uygulama programlarının daha yaygın hale gelmesi sonunda, standart kişisel bilgisayar (PC) giriş_çıkış yolunun, işlemci ile grafik arabirimleri arasındaki veri alışverişinde hız ve etkinlik açısından yetersiz kaldığı görülmüştür.

Yüksek hızlı bilgi alışverişi gerektiren giriş_çıkış birimlerinin ana işlemci yoluna daha yakın bir yapı içerisinde çalıştırılmaları, oluşan darboğazın giderilmesinde önemli rol oynamış ve bu sayede değişik yerel yol mimarileri üretilmiştir. Bu mimarilerden, PCI yerel yolu, hem 32-bit hem de 64-bit çoğullanmış (multiplexed) adres ve veri yolunu desteklemesi, donanım tasarımlarını basitleştirmesi, üretim maliyetini azaltması, grafik uygulamalarının yanısıra disk ve yerel ağ bağlantılarında da yüksek performans sağlaması sayesinde kişisel bilgisayar endüstrisinde standart haline gelmiştir.

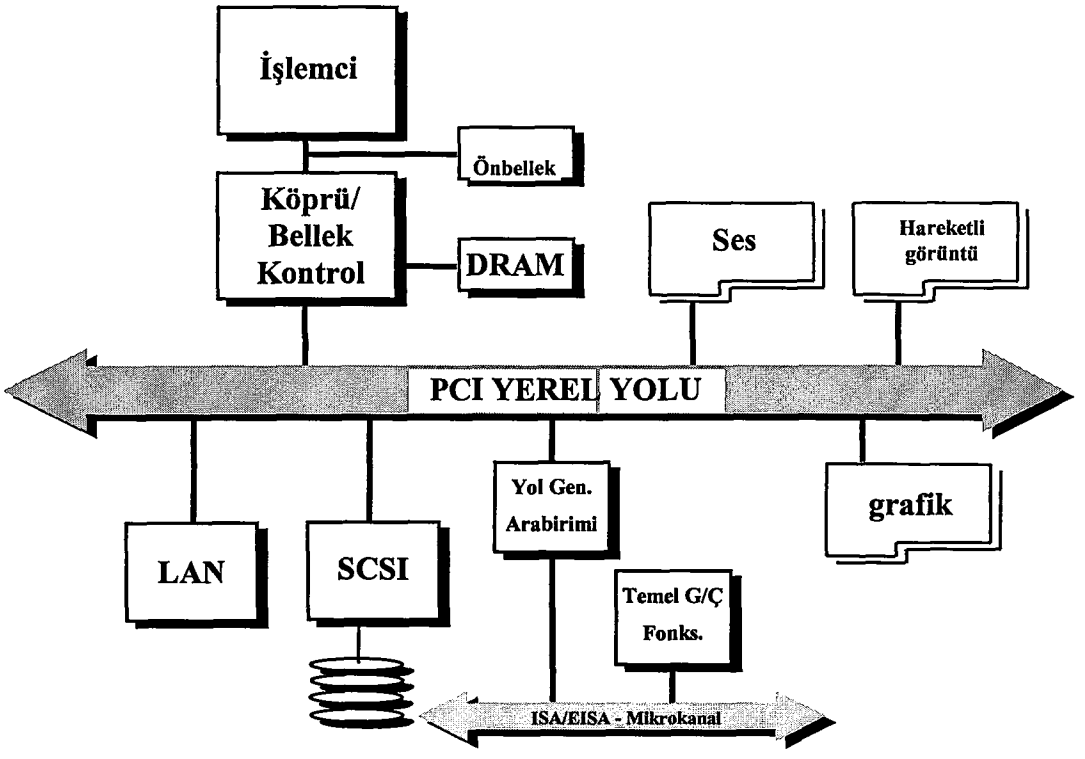
Şekil 2.1'de görüldüğü gibi PCI yerel yolu, taşınabilir sistemler ile yüksek performanslı sunucu sistemlerine uygun olmasının yanısıra, Intel x86, Dec Alpha AXP, IBM PowerPC gibi değişik işlemcileri desteklemekte ve ileride üretilebilecek işlemciler konusunda da açık ve uygulanabilir standartlar getirmektedir. PCI yerel yolunun tam otomatik yapılandırılabilmesi özelliği, 64-bit'lik yapıları desteklemesi ve hem 5V hem de 3.3V lojik seviyelerinde çalışabiliyor olması kişisel bilgisayar endüstrisi tarafından standart yapı olarak kabul edilmesinin diğer nedenleridir.

PCI yerel yolu, giriş_çıkış birimlerinin işlemciye bağlantısını, işlemcinin özelliğinden bağımsız kurallar çerçevesinde tanımladığı için, PCI uyumlu çevre birimleri, bellek modülleri, disk, yerel ağ ve grafik bağlantı birimleri, ana işlemci yapısından etkilenmeden tasarlanabilmektedirler. Bu özellik, günümüz bilgisayar sistemlerinin teknolojik gelişmelere paralel olarak üretilecek yeni ve yüksek performanslı işlemcilere daha kolay aktarılabilmesini sağlayacaktır.



Şekil 2.1 PCI yerel yolunun kullanıldığı alanlar (PCI SIG, 1995)

PCI yerel yolu etrafında kurulmuş tipik bir bilgisayar sistemi yapısını Şekil 2.2'deki gibi gösterebiliriz. Bu yapıda işlemci dışındaki birimler doğrudan PCI yerel yoluna bağlanmışlardır. İşlemci ise bir PCI köprüsü ile bir taraftan PCI yerel yoluna diğer taraftan da bellek modüllerine bağlanmıştır. Aradaki köprü, işlemcinin bellek veya giriş_çıkış adres alanına denk düşürülmüş çevre birimlerini adreslemesini ve çevre birimlerinin de belleğe erişmesini sağlamaktadır. Şekil 2.2'deki diğer bir özellik de PCI yerel yoluna farklı mimaridaki yolların bağlanmasını sağlayan Yol Genişletme Arayüzü (Bus Expansion Interface) 'dür. PCI yerel yolunun özellik ve avantajları toplu halde Tablo 2.1'de gösterilmiştir.



Şekil 2.2 PCI yerel yolu etrafında kurulu örnek bir bilgisayar sistemi (PCI SIG, 1995)

Tablo 2.1 PCI Yerel Yolunun Özellikleri ve Yararları (PCI SIG, 1995)

Yüksek performans	• 32-bit, 33 MHz'de 132 MB/sec, 66 MHz'de 264 MB/sec; 64-bit, 33 MHz'de 264 MB/sec, 66 MHz'de 528 MB/s tepe aktarım hızı • Değişken uzunlukta, lineer ve önbelleklenebilir yapıda aktarım imkanı • Düşük gecikme (latency) süresi (33 MHz'de 60-ns yazma, 66 MHz'de 30-ns yazma gecikmesi) • İşlemci, bellek arasında tam eşzamanlı (concurrent) çalışma • Senkron yol çalışması • Merkezi seçme (arbitration)
Düşük Maliyet	• Doğrudan silikon üzerinde ASIC olarak gerçekleştirilebilme imkanı • Çoğullanmış yol yapısı sayesinde düşük bacak sayısı
Kolay Kullanım	• Tamamen otomatik konfigürasyon imkanı
Geniş Kullanım Alanı	• İşlemciden bağımsız tasarım ve çalışma • 64-bit adresleme imkanı • hem 5V hem de 3.3V çalışma özelliği
Veri Güvenilirliği	• hem adres hem de veri yolu için eşlik biti

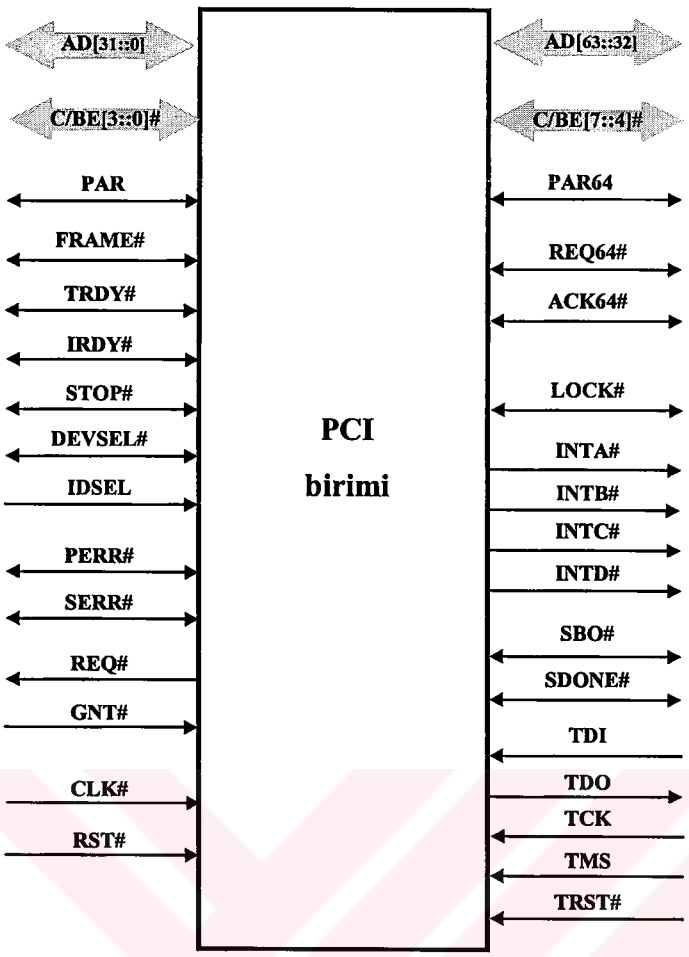
2.2 PCI Yerel Yolunu Oluşturan Sinyaller

PCI yerel yoluna bağlanan birimler, ana birim (PCI master) ve hedef birim (PCI target) olmak üzere iki sınıfa ayrılırlar. Ana birimler, yerel yola bağlı diğer birimlere erişme isteği oluşturma hakkına sahiptirler. Hedef birimler ise, ancak ana birimler tarafından adreslenip erişilebilirler. Ana birimler, hedef birim olarak da çalışabilirler.

PCI yerel yolu, adresleme, veri aktarımı, arayüz (interface) kontrolü gerçekleştirmek, değişik birimlerden gelen istekler arasında belirli bir metoda göre seçim yaparak hizmet verebilmek ve diğer sistem fonksiyonlarını yerine getirebilmek için ana birimlerde 49 sinyal, hedef birimlerde ise 47 sinyal kullanmaktadır. Bu sinyaller aralarında zorunlu ve seçime bağlı sinyaller olmak üzere iki temel sınıfa ayrılırlar. Şekil 2.3'de zorunlu sinyaller sol tarafta, seçime bağlı sinyaller ise sağ tarafta olmak üzere, PCI yerel yolunu oluşturan sinyaller gösterilmişlerdir. Sinyal isimlerindeki # sembolü, o sinyalin sıfır-aktif olduğunu gösterir. Bu sinyallerin elektriksel özellikleri Tablo 2.2'de verilmiştir.

Tablo 2.2 PCI Yerel yoluna ait sinyallerin elektriksel özellikleri

Sinyalin türü	**Sinyalin elektriksel özellikleri**
in	Sadece giriş yönünde çalışan standart sinyaldir.
out	Totem-pole yapısında çıkış yönünde çalışan standart aktif sürücü sinyaldir.
t/s	Üç durumlu, iki yönlü, yüksek empedanslı giriş/çıkış sinyalidir.
s/t/s	Aynı anda yalnızca bir birim tarafından sürülebilen, sıfır aktif üç durumlu bir sinyaldir. Bir s/t/s sinyalini lojik 0'a süren birim, bu sinyali bırakmadan en az bir sistem saati önce sinyali lojik 1'e sürmelidir. Yeni bir birim, sinyali sürebilme hakkını sinyali süren son birimden en az 1 sistem saati sonra elde eder. Herhangi bir birim tarafından sürülene kadar sinyalin aktif omayan durumda kalması için, merkezi kaynak bu sinyali lojik 1'e çekili (pull-up) durumda tutar.
o/d	Bu tip sinyaller, birden fazla birim tarafından telli veya (wire-OR) fonksiyonu gerçekleşecek şekilde paylaşılırlar. Herhangi bir birim tarafından sürülmediği zaman aktif olmayan durumda kalabilmesi için bu sinyal merkezi kaynak tarafından lojik 1'e çekili tutulur.



Şekil 2.3 PCI yerel yolunu oluşturan sinyaller

2.2.1 PCI yerel yolundaki sistem sinyalleri

CLK (Clock) (in) sinyali, PCI yerel yolunda gerçekleşen tüm işlemler için zamanlama bilgisi sağlar. RST#, INTA#, INTB#, INTC# ve INTD# dışındaki PCI sinyalleri CLK sinyalinin yükselen kenarı ile örneklenirler. Diğer zamanlama parametreleri de bu yükselen kenara göre tanımlanırlar. PCI yerel yolu, 0 Hz'den yerel yolun tipine bağlı olarak maksimum 33 MHz veya 66 MHz'e kadar olan sistem saatleri ile çalışabilir.

RST# (Reset#) (in) sinyali, PCI yerel yoluna özgü saklayıcıları ve sinyalleri açılış durumuna getirmek için kullanılır. RST# sinyalinin aktif olması ile çıkış olarak tanımlanmış tüm PCI sinyalleri aktif olmayan durumlarına çekilirler. Bu sinyal, CLK sinyalinden bağımsız olarak herhangi bir anda aktif hale getirilebilir.

2.2.2 PCI yerel yolundaki adres ve veri sinyalleri

PCI yerel yolunda adres ve veri bilgileri aynı fiziksel sinyaller kullanılarak çoğullanma yöntemi ile aktarılırlar. Yol üzerindeki işlemler, bir adres aşaması ve takip eden bir veya daha fazla veri aşamasından oluşur. Adres aşaması, FRAME# sinyalinin aktif olduğu sistem çevrimi olarak tanımlanmıştır. Adres aşamasında AD[31::00], 32 bitlik fiziksel adresi içerir. Bu adres, giriş_çıkış için byte adresi, konfigürasyon ve bellek erişimleri için DWORD (4 byte) adresi olarak değerlendirilir. Veri aşamasında AD[07::00] alçak anlamlı byte'ı, AD[31::24] ise yüksek anlamlı byte'ı içerir.

PCI komutları ve byte seçme sinyalleri çoğullanma yöntemi ile aynı PCI sinyalleri kullanılarak aktarılırlar. C/BE[3::00]# (t/s) sinyalleri, adres aşamasında PCI komutunu tanımlar, veri aşamasında ise aynı DWORD içindeki hangi byte'lara erişileceğini belirtir. Byte seçme sinyalleri veri aşamasının tamamında geçerlidir ve hangi byte'ların anlamlı olduğunu belirler. Bir DWORD'u oluşturan 4 byte'tan alçak anlamlısı C/BE[0]# , yüksek anlamlısı C/BE[3]# ile seçilir.

Bütün PCI birimleri AD[31::00] ile, C/BE[3::00]# sinyallerini kapsayacak şekilde çift eşlik biti üretmek zorundadırlar. Adres aktarımında PAR (t/s) sinyali, adres aşamasından bir sistem saati sonra kararlı ve geçerli olur. Veri aktarımında ise PAR sinyali, yazma için IRDY# , okuma için TRDY# sinyali aktif olduktan bir sistem saati sonra kararlı ve geçerli hale gelir. PAR sinyali geçerliliğini yürütülmekte olan aşamadan bir sistem saati sonrasına kadar korur. Ana birim PAR sinyalini adres ve yazma yönündeki veri aşamaları için, hedef birim ise okuma yönündeki veri aşamaları için üretir.

2.2.3 PCI yerel yolundaki arayüz kontrol sinyalleri

FRAME# (s/t/s) sinyali, o an PCI yerel yolunu kullanan ana birim tarafından erişimin başlangıcını ve süresini göstermek için sürülür. FRAME# sinyali aktif olduğu anda yerel yolun üzerinde bir aktarımın başladığı belirtilir ve bu sinyal aktif olduğu sürece veri

aktarımı devam eder. FRAME# sinyali geri çekildiğinde ya son veri aşaması devam etmektedir ya da veri aşaması sona ermiştir.

IRDY# (Initiator Ready) (s/t/s) sinyali PCI yerel yolunda veri aktarımını başlatan birimin o anki veri aşamasını tamamlayabileceğini gösterir. IRDY# sinyali TRDY# sinyali ile birlikte kullanılır. Veri aşaması, IRDY# ve TRDY# sinyallerinin ikisinin de aktif olduğu ilk sistem saatinde tamamlanabilir. IRDY# sinyali, yazma sırasında geçerli verinin AD[31::00]'da olduğunu, okuma sırasında ise ana birimin veri almaya hazır olduğunu gösterir. IRDY# ve TRDY# sinyalleri birlikte aktif edilene kadar araya bekleme çevrimleri eklenir.

STOP# (s/t/s) sinyali ile, aktif olan hedef birim o anda gerçekleşmekte olan işlemin durdurulmasını işlemi başlatan ana birimden ister.

LOCK# (s/t/s) sinyali bölünemez bir işlemin tamamlanması için yerel yol erişimi yapılacağını gösterir. LOCK# sinyali aktif edildiği andaki adres aşamasında belirtilen adres bölgesine kesintisiz ve özel erişim yapılacağını belirttiği için, diğer birimler ancak LOCK# sinyali ile kilitlenmemiş adreslere erişebilirler. PCI yerel yoluna erişim hakkı verilmesi, hakkı elde eden birimin aynı zamanda LOCK# sinyalini de kontrol edebileceğini garanti edemez. LOCK# sinyalinin kullanımı GNT# sinyali ile birlikte özel bir protokolle gerçekleştirilir.

IDSEL (ID Select) (in) sinyali konfigürasyon ile ilgili okuma ve yazma işlemleri sırasında birimleri seçmek için kullanılır.

DEVSEL# (Device Select) (s/t/s) sinyali, adres aşamasında aktarılan adresin kendi adresi olduğunu fark eden PCI hedef birimi tarafından adresleme işlemini tamamlamak için aktif hale getirilir. Bir adresleme işleminde DEVSEL# sinyalinin aktif hale gelmemesi, o adrese atanmış bir birimin olmadığını belirtir.

TRDY# (Target Ready) (s/t/s) sinyali PCI yerel yolunda gerçekleşmekte olan veri aktarımına ait hedef birimin yürütülmekte olan veri aşamasını tamamlayabileceğini

gösterir. TRDY# ve IRDY# sinyalleri birlikte kulanılır. Hem TRDY# hem de IRDY# sinyallerinin aynı anda aktif olduğu ilk sistem saatinde veri aktarımı gerçekleştirilir. Okuma aşamasında TRDY# sinyali, AD[31::0] üzerinde geçerli verinin olduğunu, yazma aşamasında ise hedef birimin veriyi kabul etmeye hazır olduğunu gösterir.

2.2.4 PCI yerel yolundaki seçme sinyalleri

REQ# (Request) (t/s) sinyali, seçiciye (arbiter) bu sinyali aktif hale getiren birimin yerel yolu kullanmak istediğini bildirir. REQ# sinyali noktadan noktaya bir sinyaldir. Her ana birimin, RST# sinyali aktif olduğunda üç durumlu, yüksek empedanslı konuma geçen ayrı bir REQ# sinyali vardır.

GNT# (Grant) (t/s) sinyali, REQ# sinyali ile istekte bulunmuş olan birime yola erişim hakkının verildiğini bildirir. GNT# sinyali de noktadan noktaya bir sinyaldir.

2.2.5 PCI yerel yolundaki hata bildirim sinyalleri

PERR# (Parity Error) (s/t/s) sinyali, özel çevrim (special cycle) dışındaki bütün PCI işlemleri sırasında oluşabilecek veri eşlik hatalarını bildirir. Veri eşlik hatası oluştuğunda hatayı belirleyen birim veriyi aldıktan iki sistem saati sonra PERR# sinyalini aktif hale getirir. PERR# sinyali veri eşlik hatası olan her veri aşaması için en az bir sistem saati sürer. Eğer ardışık veri aşamalarında veri eşlik hatası varsa PERR# sinyali bir sistem saatinden uzun sürer. Veri eşlik hatasının kaybolması veya bildirilmesinin gecikmesi durumlarında özel bir işlem yapılmaz.

SERR# (System Error) (o/d) sinyali özel çevrimdeki veri eşlik hataları ile adres eşlik hatalarını ve sonucu sistemin çalışmaya devam etmesini engelleyecek nitelikte önemli olan diğer sistem hatalarını bildirmek için kullanılır.

2.2.6 PCI yerel yolundaki kesme sinyalleri

PCI yerel yolunda INTA# (Interrupt A) (o/d), INTB# (Interrupt B) (o/d), INTC# (Interrupt C) (o/d) ve INTD# (Interrupt D) (o/d) olmak üzere 4 adet kesme sinyali tanımlanmıştır. Bu kesme sinyalleri seçimlik ve seviye duyarlı (level sensitive) sinyallerdir. Herhangi bir INTx# sinyali aktif hale getirildiğinde, yazılım tarafından bu kesmeye servis verilene kadar aktif durumda kalır. Yazılım kesme isteğini servis etmeye başladığında, INTx# sinyali geri çekilir. PCI yerel yoluna bağlanan birimler sadece bir fonksiyon yerine getiriyorlarsa, bu birimlerin kullanımına sadece INTA# sinyali ayrılır. Eğer aynı birim içerisinde birden fazla bağımsız fonksiyon gerçekleniyor ve herbiri ayrı ayrı kesme ihtiyacı duyuyorsa INTA# sinyaline ek olarak INTD# 'ye kadar olan sinyaller de kullanılır.

2.2.7 PCI yerel yolundaki önbellek destek sinyalleri

Önbelleklenebilir PCI bellek modüllerinin, write-through veya write-back yöntemlerine göre çalışabilmesi için SBO# (in/out) ve SDONE (in/out) önbellek destek sinyalleri tanımlanmıştır. SBO# (in/out) sinyalinin aktif edilmesi önbellek içerisinde içeriği değiştirilmiş bir alana erişildiği belirtilir. SDONE (in/out) sinyali aktif edilmesi önbelleğe erişimin hala devam ettiğini ve sonucun beklendiğini gösterir.

2.2.8 PCI yerel yolundaki 64-bit genişleme sinyalleri

AD[63::32] (Address) (t/s) hatları üzerinde, adres ve veri bilgisi çoğullanması ile ek 32 bit sağlanır. Adres aşamasında DAC (çift adres çevrimi) komutu kullanıldığında ve REQ64# sinyali aktif edildiğinde 64 bitlik adresin yüksek anlamlı 32 biti bu sinyaller tarafından aktarılır.

C/BE[7::4]# (Command / Byte Enable) (t/s) sinyalleri kullanılarak PCI komutları ve byte seçme sinyalleri çoğullama yöntemi ile aktarılırlar. Adres aşamasında yol komutu C/BE[7::4]# üzerinden aktarılır. Veri aşamasında C/BE[7::4]# sinyalleri, aynı DWORD içindeki hangi byte'ların seçildiğini belirtirler.

REQ64# (Request 64) (s/t/s) sinyali PCI yerel yolunu kullanan ana birim tarafından 64 bitlik veri aktarımı yapılacağını belirtmek için FRAME# sinyali ile aynı zamanlamayla aktif hale getirilir.

ACK64# (Acknowledgement 64) (s/t/s) sinyali hedeflenen PCI biriminin 64 bitle veri aktarımı yapmayı kabul ettiğini gösterir. ACK64# sinyalinin zamanlaması DEVSEL# sinyali ile aynıdır.

PAR64 (Parity 64) (t/s) sinyali AD[63::32] ve C/BE[7::4]# sinyalleri için üretilen çift eşlik bitidir. PAR64 sinyalinin zamanlaması PAR sinyali ile aynıdır. Ana birimler PAR64# sinyalini adres ve yazma aşamaları için, hedef birimler ise okuma aşamaları için kullanırlar.

2.2.9 PCI yerel yolundaki diğer sinyaller

PRSNT[1:2]# (Present) (in) sinyalleri PCI yerel yoluna fazladan birim bağlanabilmesi için ilave kart olup olmadığını, eğer varsa bu kart için gerekli toplam güç ihtiyacının ne olduğunu belirtirler.

CLKRUN# (Clock Run) (in, o/d, s/t/s) sinyali PCI yerel yoluna bağlı birimlere giriş bilgisi olarak CLK sinyalinin durumunu bildirdiği gibi herhangi bir birim tarafından sistem saatinin başlatılmasını veya hızının arttırılmasını sağlamak için de kullanılır.

2.3 PCI Yerel Yolunda Tanımlı Olan Komutlar

PCI yerel yol komutları, hedef birimlere ana birimlerin isteklerini bildirmek için kullanılır. Tablo 2.3'de görülen bu komutlar, adres aşaması sırasında C/BE[3::0]# sinyalleri aracılığı ile hedef birimlere aktarılırlar.

Tablo 2.3 PCI yerel yol komutlarının bit karşılıkları ve isimleri (PCI SIG, 1995)

C/BE[3::0]#	Komutun Adı
0000	Interrupt Acknowledge
0001	Special Cycle
0010	I/O Read
0011	I/O Write
0100	-
0101	-
0110	Memory Read
0111	Memory Write
1000	-
1001	-
1010	Configuration Read
1011	Configuration Write
1100	Memory Read Multiple
1101	Dual Address Cycle
1110	Memory Read Line
1111	Memory Write and Invalidate

Interrupt Acknowledge komutu, dolaylı olarak sistem kesme kontrol biriminden okuma komutuna dönüştürülür. Adres aşamasında PCI yerel yoluna çıkarılan adresler anlam ifade etmezler. Byte seçme sinyalleri ise kaç byte'lık bir kesme vektörü döndürülmesi gerektiğini belirtir.

Special Cycle komutu, PCI yerel yolu üzerinde basit mesaj yayınına imkan sağlayan bir mekanizmadır. PCI veri yolu üzerinden tanımlı standart komutlar dışında uygulamaya yönelik haberleşme yapılabilmesini sağlar.

I/O Read komutu, giriş_çıkış adres alanına denk düşürülmüş birimlerden veri okumak için kullanılan komuttur. Erişilmek istenen giriş_çıkış adresi AD[31::00] sinyalleri ile byte çözünürlüğünde kodlanmıştır. Bu nedenle 32 bitlik adresin tamamı değerlendirilmelidir.

Memory Read komutu, bellek adres alanına denk düşürülmüş hedef birimlerden veri okumak için kullanılır. Eğer okunan veri ile ilgili tutarlılık problemi oluşmayacağı garanti edilebiliyorsa performansı arttırmak amacıyla o an için ihtiyaç duyulmayan veriler de okunabilir. Herhangi bir senkronizasyon işleminden önce sistemin kararlılığını sağlayabilmek için bu bilgilerin tutulduğu arabelleklerin içerikleri geçersiz kılınmalıdır.

Memory Write komutu, bellek adres alanına denk düşürülmüş hedef birimlere veri yazmak için kullanılır. Hedef birim yazma işlemi için hazır olduğunu bildirirse, yazma sırasında kendisine aktarılan verinin sıralı ve tutarlı yazılmasını garanti etmelidir. Bunun için ya komutun tamamen senkron yürütülmesi ya da okuma işlemlerinden önce yazılıma saydam bir şekilde arabelleklerin temizlenmesi sağlanmalıdır.

Configuration Read komutu, PCI yerel yolu üzerinde yer alan birimlerin konfigürasyon alanlarını okumak için kullanılır. Bu birimler, konfigürasyon erişimi için IDSEL sinyali aktif hale getirilerek ve AD[1::0] sinyalleri "00" yapılarak seçilirler. AD[7::2] sinyalleri ile konfigürasyon alanındaki 64 adet DWORD saklayıcısından hangisine erişileceği belirtilir. AD[31::11] sinyalleri anlamlı bilgi içermezler. AD[10::8] sinyalleri ise çok fonksiyonlu hedef birimlerde, hangi fonksiyona ait bölümün adreslendiğini gösterirler.

Configuration Write komutu, PCI birimlerinin konfigürasyon alanlarına veri aktarmak için kullanılır. Bu birimler, IDSEL sinyali aktif hale getirilip AD[1::0] sinyalleri "00" yapılarak seçilirler. Konfigürasyon çevriminin adres aşamasında, her birimin konfigürasyon alanındaki 64 adet DWORD saklayıcısından hangisinin seçileceği AD[7::2] sinyalleri ile belirtilir. AD[31::11] sinyalleri anlamlı bilgi taşımazlar ve AD[10::8] sinyalleri çok fonksiyonlu birimlerde hangi fonksiyonun adreslendiğini gösterirler.

Memory Read Multiple komutunun çalışma mantığı Memory Read komutu ile aynıdır fakat ana birimin bağlantıyı kesmeden önce birden fazla önbellek satırı okuyabileceğini gösterir. Bellek kontrolcüsü, FRAME# sinyali aktif olduğu sürece bellekten okuma isteklerini işhattı mantığı ile yerine getirir. Bu komut bir veya birden fazla önbellek satırından ardışık okumayı desteklediği için performansı arttırmak amacıyla çok sayıda ardışık veri transferi yapılmasını gerektiren durumlarda tercih edilir.

Dual Address Cycle komutu, erişilmek istenen adresin ilk 4 GB'lık adres alanında olmadığı durumlarda 64 bit'lik adres aktarmak için kullanılır.

Memory Read Line komutu, Memory Read ile aynı yapıdadır, fakat ana birimin hedef birimden bir önbellek satırının tamamını alabileceğini gösterir. Bu komut, çok sayıda

ardışık verinin aktarımında , bellek sisteminin her istek için ayrı bir bellek çevrimi yapması yerine bütün bir önbellek satırını bir seferde okuyarak değerlendirmesi sebebiyle performansı arttırmaktadır. Bellek okuma komutunda olduğu gibi herhangi bir senkronizasyon işleminden önce, önceden tutulmuş ara bellekler geçersiz kılınır.

Memory Write and Invalidate komutunun çalışma prensibi Memory Write komutunun aynısıdır. Farklı olarak en küçük aktarım birimi bir önbellek satırıdır. Ana birim hedef tarafından kesme gelmediği takdirde adreslenmiş önbellek satırına bütün bilgiyi tek bir işlemde yazar. Bu komut için her veri aşamasında bütün byte seçiciler aktif olmalıdır. Bu komutun doğru çalışması ana birimdeki önbellek satır uzunluğunu gösteren konfigürasyon saklayıcısının gerçeklenmiş olmasına bağlıdır.

Kullanıma açık olmayan (reserved) PCI komutları ileride doğabilecek ihtiyaçları karşılamak içindir. Hedef birimler bu komutlara cevap vermek zorunda değildirler. PCI yolu üzerinde bu komutlar kullanıldığında genellikle Master-Abort ile geri çevrilirler.

2.4 PCI Yerel Yolunda Veri Aktarımı

PCI yerel yolu üzerinde veri aktarımı temel olarak kısa ve anlık işlemlerle gerçekleştirilir. Kısa ve anlık aktarımlar hem bellek hem de giriş_çıkış adres alanlarında desteklenirler. Veri aktarımında kullanılan sinyaller için sistem saatinin yükselen kenarına göre en geç ne kadar önce aktif olmaları gerektiğini ve en az ne kadar süreyle aktif kalmaları gerektiğini belirten bir zamanlama vardır. Bu zaman içerisinde sinyallerde lojik seviyeler arasında geçişlere izin verilmez. RST#, INTA#, INTB#, INTC# ve INTD# sinyalleri sistem saatinden bağımsız oldukları için böyle bir zaman kısıtlamaları yoktur.

Veri aktarımı özellikle FRAME#, IRDY# ve TRDY# sinyaleri kullanılarak gerçekleştirilir. FRAME# sinyali, ana birim tarafından bir aktarımın başlangıcını ve sonunu belirlemek için, IRDY# sinyali ise veri transferine hazır olunduğunu göstermek için kullanılır. TRDY# sinyali, hedef birimin veri transferine hazır olduğunu belirtir.

FRAME# ve IRDY# sinyalleri aktif değilken PCI yerel yolu boş bekleme (idle) konumundadır. FRAME# sinyalinin aktif edildiği ilk sistem saati adres aşamasıdır. Adres ve yol komutları bu saatin yükselen kenarında aktarılırlar. Bir sonraki sistem saati ana birim ile hedef birim arasında veri aktarımını başlatır. IRDY# veya TRDY# sinyallerinin aktif olmadığı zamanlarda veri aşamasına ana birim veya hedef birim tarafından bekleme çevrimleri (wait cycle) eklenir. Aktarılacak veri geçerli ise verinin kaynağı, yazma için IRDY#, okuma için TRDY# sinyalini aktif hale getirir. Veriyi alacak birim henüz hazır değilse kendi xRDY# sinyalinin aktif etmez. Veri, IRDY# ve TRDY# sinyallerinin her ikisinin de sistem saatinin aynı yükselen kenarında aktif olması durumunda aktarılır.

Ana birim IRDY# sinyalini aktif hale getirirse, yürütülmekte olan veri aşaması tamamlanana kadar TRDY# sinyali ne olursa olsun IRDY# ve FRAME# sinyallerinin durumlarını değiştiremez. Hedef birim TRDY# veya STOP# sinyallerini aktif ederse o anki veri aşaması tamamlanana kadar DEVSEL#, TRDY# veya STOP# sinyallerinin durumunu değiştiremez. Ana birim veya hedef birim başlattığı veri aktarımını o anki veri aşaması tamamlanana kadar sürdürmelidir. Veri aşaması IRDY# ve TRDY# veya STOP# sinyalleri aktif edilirse tamamlanır. Verinin aktarılıp aktarılmayacağı TRDY# sinyalinin durumuna bağlıdır.

Ana birim sadece bir veri aktarım çevrimi gerçekleştirmek istediğini göstermek için FRAME# sinyalini geri çekerken IRDY# sinyalini aktif hale getirir. Hedef birim TRDY# sinyalini aktif ederek son veri transferini tamamlamaya hazır olduğunu bildirdikten sonra PCI arabirim kontrolcüsü FRAME# ve IRDY# sinyallerini geri çekerek boş bekleme durumuna geçer.

PCI yerel yolu üzerinde veri aktarımını optimize etmek için bir işlem daha büyük bir işlemin alt parçası haline getirilebilir. Bunu gerçekleştirmek için toplama, birleştirme veya eleme işlemleri yapılabilir.

Toplama işleminde, ardışık bellek yazma işlemleri, lineer anlık_aktarım sıralaması kullanılarak tek bir PCI yerel yol işleminde toplanırlar.

Byte birleştirme işleminde, ardarda gelen ve birbirlerinden bağımsız olan byte veya word uzunluğundaki bellek yazma işlemleri tek bir DWORD'da birleştirilirler.

Önbellek satırı birleştirme işlemi, ardışık bellek yazma işlemlerinin bir önbellek satırında birleştirilmesidir. Önbellek satırı birleştirme, adres alanı önbelleklenebilir tanımlandığında veya önbellek satırı veri aktarımı için byte birleştirme ve/veya toplama işlemi kullanıldığında yapılabilir.

Eleme işlemi ise aynı bellek bölgesine yapılan birden fazla yazma işlemini tek bir yol işlemine dönüştürür.

2.5 PCI Yerel Yolunda Adreslerin Çözülmesi

PCI yerel yolu, bellek adres alanı, giriş_çıkış adres alanı ve konfigürasyon adres alanı olmak üzere üç fiziksel adres alanı tanımlar. Bellek ve giriş_çıkış adres alanları uygulamaların kullanımı için tanımlanmıştır. Konfigürasyon adres alanı ise PCI donanım konfigürasyonunu desteklemek amacıyla oluşturulmuştur.

PCI yerel yolu ile ana sistem arasında bağlantıyı sağlayan yol köprüsü haricindeki PCI hedef birimleri, iç saklayıcılarına veya sundukları fonksiyonlara erişimi sağlayabilmek için taban adres saklayıcılarına sahip olmalıdırlar. Konfigürasyon yazılımı, taban adres saklayıcısını herhangi bir PCI biriminin verilen adres alanında ne kadar yere ihtiyacı olduğunu ve bu alanda hangi adrese yerleşeceğini belirlemek için kullanılır.

PCI yerel yolunda adres kodu çözme işlemi dağıtılarak yapılır. Her birim kendi adresinin kodunu çözer. PCI yerel yolu pozitif kod çözme ve negatif kod çözme yöntemlerini destekler. Pozitif kod çözme işlemi, her birim kendisine atanan adrese erişilip erişilmediğini kontrol ettiği için daha hızlıdır. Negatif kod çözme aynı anda sadece bir birim tarafından uygulanabilir, çünkü negatif kod çözmeyi uygulayan birim ancak diğer birimler tarafından pozitif kod çözme ile kodu çözülmemiş erişimleri kabul eder. Bu sebeple negatif kod çözme daha yavaştır. Buna rağmen PCI yerel yolunu genişletmek

amacıyla kullanılan birimler, adres alanında nereye yerleşeceklerini belirlemek için negatif kod çözme yöntemini uygulayabilirler.

Bir hedef birim, erişilmek istenen adres bölgesine sahipse DEVSEL# sinyalini aktif hale getirir. Kod çözme için kullanılacak AD hatlarının adedi hedef birimin kapladığı adres alanının uzunluğuna bağlıdır. Örneğin 4 byte kaplayan bir birim, AD[1::0]'ın kodunu çözmeye ihtiyaç duymaz. Erişim isteği byte seçme sinyallerinin değeri göz önüne alınarak tamamlanır. PCI yerel yolu birimlerin bir DWORD'u başka birimler ile paylaşmalarına izin verir. Bu birimler adreslendiklerinde byte seçme sinyallerinin gösterdiği byte'ların kendilerine ait olup olmadığını kontrol ederek cevap verirler.

Konfigürasyon adres alanında erişilecek adresler, AD[7::2] sinyalleri kullanılarak DWORD çözünürlüğünde belirlenirler. Hedef birimler konfigürasyon erişiminin kendilerine yapıldığını IDSEL sinyali aktif ve AD[1::0] sinyalleri "00" değerine sahipken geçerli bir konfigürasyon komutu çözdüklerinde anlarlar ve DEVSEL# sinyalini aktif hale getirirler.

2.6 PCI Yerel Yolunda Okuma İşlemi

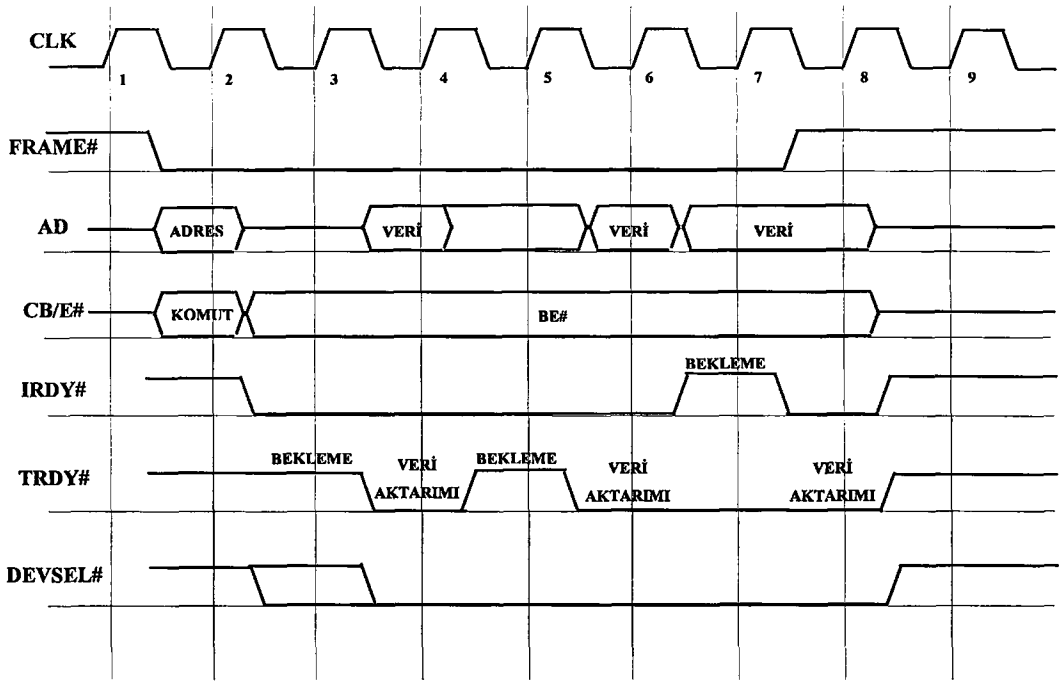
Şekil 2.4'de PCI yerel yolunda gerçekleşen bir okuma işlemi görülmektedir. Yerel yola erişim FRAME# sinyali aktif olduğu sürece devam eder. Adres aşamasında AD[31::00] sinyalleri adres bilgisini, C/BE[3::00]# sinyalleri ise PCI komutunu içerir.

İlk veri aşaması erişim başladıktan sonra 3. periyodda gerçekleşir. Veri aşamasında, C/BE[3::00]# sinyalleri hangi byte'ların seçildiğini belirtirler. C/BE[3::00]# sinyalleri veri aşaması süresince IRDY# sinyalinin durumundan bağımsız olarak geçerli byte seçme bilgisini içerirler. Veri aşamaları ya bekleme çevriminden ya da veri aktarımı işlemlerinden oluşur. C/BE[3::00]# sinyalleri okuma ve yazma işlemlerinde veri aşamasının ilk sistem saatinden başlayarak son işleme kadar geçerli olurlar. N. veri aşamasının tamamlanmasından sonraki periyodda C/BE[3::00]# sinyalleri (N+1). veri aşamasının byte seçme bilgilerini tutarlar. IRDY# veya TRDY# sinyallerinden herhangi biri aktif değilken bekleme çevrimine geçilir ve veri aktarımı yapılamaz. Şekil 2.4'de 4., 6. ve 8.

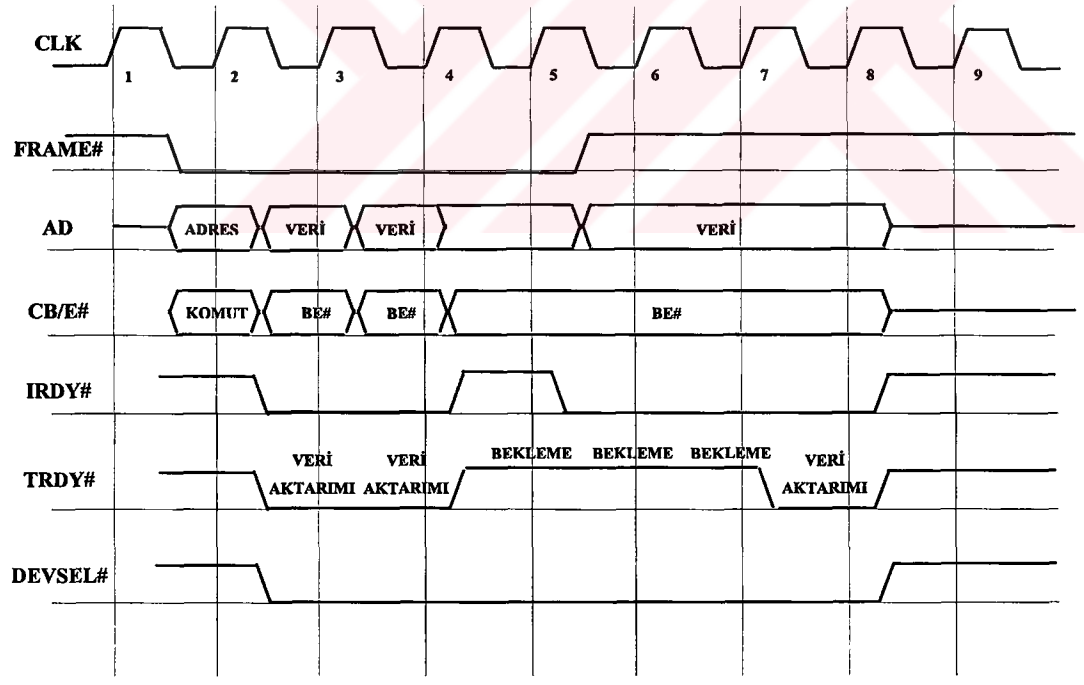
periyodlarda veri aktarılmakta, 3., 5. ve 7. periyodlarda ise bekleme çevrimi uygulanmaktadır. Ana birim 7. periyodda bir sonraki veri aşamasının son veri aşaması olduğunu bilmektedir. Fakat son veri aktarımını tamamlamaya hazır olmadığı için IRDY# sinyali, 7. periyodda geri çekilmiş ve FRAME# sinyali aktif olarak kalmıştır. 8. periyodda IRDY# sinyali aktif edildiğinde FRAME# sinyali geri çekilerek hedef birime son veri aşaması olduğu bildirilir.

2.7 PCI Yerel Yolunda Yazma İşlemi

PCI yerel yolu üzerinde gerçekleşen örnek bir yazma işlemi, Şekil 2.5'de verilmiştir. İşlem, FRAME# sinyalinin aktif hale getirildiği 2. periyodda başlamaktadır. Yazma işlemi, yapı olarak okuma işlemi ile aynıdır. Fakat yazma işleminde ana birim hem adres hem de veriyi sağladığı için adres aşamasını takip eden bir duraklama çevrimine ihtiyaç yoktur. Şekil 2.5'de 1. ve 2. veri aşamaları bekleme çevrimi olmaksızın tamamlanmışlardır. Bir sonraki veri aşamasında ise hedef birim tarafından oluşturulmuş üç bekleme çevrimi vardır. FRAME# sinyali son veri aşamasını göstermek için geri çekildiğinde IRDY# sinyali aktif hale getirilmektedir. 5. periyodda ana birim tarafından IRDY# sinyali geri çekilerek veri aktarımı geciktirilmektedir. Son veri aşaması ise 6. periyodda başlayıp 8. periyodda tamamlanmaktadır.



Şekil 2.4 PCI Yerel yolunda okuma işleminin gerçekleşmesi (PCI SIG, 1995)



Şekil 2.5 PCI Yerel yolunda yazma işleminin gerçekleşmesi (PCI SIG, 1995)

2.8 PCI Yerel Yolunda İşlemlerin Sonlandırılması ve Sıralanması

PCI yerel yolunda yürütülen işlemlerin sonladırılması ana birim veya hedef birim tarafından başlatılabilir. Her iki birimin de yürütülmekte olan işlemi tek yönlü olarak sonlandırma hakkı olmasa da ana birim esas sorumlu olarak bütün işlemlerin sıralı ve düzenli bir şekilde sonlanmasını sağlar.

Ana birim sonlandırmayı, ya yürütülen işlem düzgün olarak tamamlandığı ya da o işleme ayrılan toplam süre dolduğunda başlatır. Sonlandırma, FRAME# sinyali geri çekilip, IRDY# sinyali aktif edilerek belirtilir ve bu durum, hedef birime son veri aşamasının yürütülmekte olduğunu gösterir. FRAME# ve IRDY# sinyalleri geri çekildiğinde işlem sonlandırılmış olur. Son veri aktarımı, IRDY# ve TRDY# sinyalleri birlikte aktif iken yapılır.

Hedef birimler, ana birimden gelen isteği tamamlayamayacaklarsa STOP# sinyalini aktif ederek sonladırmanın başlatılmasını isterler. Hedef birim kaynaklı sonlardırmada üç farklı durum söz konusudur.

Retry durumu, hedef birimin geçici meşguliyetini ve veri aktarımı yapmadan ana birim ile bağlantısını sonlandırmak istediğini gösterir.

Disconnect durumu, hedef birim içerisinde anlık veri aktarımında kullanılan kaynağın sınırlarının aşılabileceğini veya kaynaklara erişimde bir çakışma olabileceğini belirtmek için kullanılır.

Target-Abort durumunda ise hedef birim, ana birime yürütülmekte olan işlemi, kendisinde oluşan düzeltilemez bir hata sebebiyle sonlandırmak istediğini belirtir.

PCI yerel yolunda bir işlemin diğer işlemlere göre sırasının nasıl değerlendirileceği ancak işlem tamamlandıktan sonra belirlenebilir. Retry ile terkedilen işlemler veri transferi yapılmadığı için henüz tamamlanmış sayılmazlar ve bu sebeple işlemlere bağlı bir sıralama ihtiyacı söz konusu olmaz. Master-Abort veya Target-Abort ile terkedilen işlemler

tamamlanmış kabul edilirler. Yeni gelen istekler ile Retry ile terkedilmiş istekler aralarında herhangi bir öncelik gözetilmeden çalıştırılabilirler. Eğer ana birim bir işlem tamamlanmadan bir diğerinin başlamasını istemiyorsa ikinci işlemi birincisini bitirmeden başlatmaz.

Yazma işlemleri üretici-tüketici (producer-consumer) modelinde tanımlanmış olan yazma kurallarına uygun gerçekleştirilir. Buna göre, bir ana birimin (üretici) gerçekleştirdiği yazma işlemlerinin sonuçları sistem içerisinde yer alan diğer ana birimler (tüketici) tarafından gerçekleştirildikleri sırada görülür. Başarımı arttırmak amacıyla uygun koşullar sağlandığında işlemler gecikmeli (posted) yapılabilirler. Bu özelliklerin sağlanabilmesi için arabelleklerin temizlenmesi gerektiğinde sistemde "deadlock" oluşması engellenmelidir.

PCI yerel yolu üzerindeki işlemlerin gecikmeli gerçekleştirilmesi, özellikle birden fazla PCI yerel yolunun birbirine köprüler ile bağlı olduğu sistemlerde önem kazanır. Gecikmeli işlemler hedef yolda tamamlanmadan önce işlemin başladığı yolda tamamlanırlar. İşlemin başladığı yol ile tamamlanacağı yol arasındaki köprü birimi veriyi gerçek hedef birim adına alır. PCI yerel yolu üzerinde bellek yazma komutları olan Memory Write, Memory Write and Invalidate komutları gecikmeli gerçekleştirilirler. Gecikmesiz işlemler ise başlangıç yolunda tamamlanmadan önce hedef yolunda tamamlanan komutlardır. Bellek okuma komutları (Memory Read, Memory Read Line, Memory Read Multiple), giriş_çıkış işlemleri (I/O Read, I/O Write) ve konfigürasyon komutları (Configuration Read, Configuration Write) gecikmesiz komutlardır.

Köprüler bellek yazma işlemlerini her iki yöne doğru da gerçekleştirebilirler. Veri tutarlılığının sağlanması için köprüde aynı yönde gerçekleştirilen gecikmeli bellek yazma işlemleri, hedef yol üzerinde de işlemlerin başladığı yol ile aynı sırada tamamlanmalıdırlar. Bunun yanı sıra her iki yönde de okuma işlemleri bitmeden önce geciktirilmiş yazma işlemleri tamamlanmalı ve önceden gecikmesiz bellek yazma işlemi başlatmış ana birimlerin, bu işlem tamamlanmadan gecikmeli bir başka yazma işleminin hedefi olmaları durumunda bu işlemi kabul etmemeleri sağlanmalıdır.

2.9 PCI Yerel Yolunda Konfigürasyon Alanı

PCI yerel yoluna bağlanan birimlerin, sistem adres alanı içerisinde hangi bölgeye yerleştirilecekleri, hangi kesme sinyallerini kullanacakları gibi işlemlerin, kullanıcıların müdahalesine gerek kalmadan sistem tarafından otomatik gerçekleştirilmesi için her PCI biriminde bir konfigürasyon alanı tanımlanmıştır. Çok fonksiyonlu PCI birimlerinde her fonksiyon için ayrı bir konfigürasyon alanı vardır. 64 x 32 bit düzeninde 256 byte'tan oluşan konfigürasyon alanının yapısı Şekil 2.6'da verilmiştir.

31	16	15	00
Birim Tanımlayıcı Kodu		Üretici Tanımlayıcı Kodu	
Durum Bilgisi		Komut	
Birim Sınıf Kodu			Üretim Kodu
Açılış Test Sonucu	Başlık Tipi	Gecikme Süresi	Önbellek Satır Uzunluğu
Taban Adres Saklayıcıları			
CardBus CIS göstergesi			
Alt Sistem Tanımlayıcı Kodu		Alt Sistem Tanımlayıcı Kodu	
Genişleme ROM Taban Adresi			
İlerisi İçin Saklı			
İlerisi İçin Saklı			
Maksimum Gecikme	Min. Kabul Süresi	Kesme Sinyali	Kesme Hattı

Şekil 2.6 PCI birimlerine ait konfigürasyon alanının yapısı (PCI SIG, 1995)

Birim Tanımlayıcı Kodu (Device ID), PCI yerel yoluna bağlanan birimi tanımlayan ve üretici firma tarafından belirlenen 16 bitlik bilgiyi içermektedir.

Üretici Tanımlayıcı Kodu (Vendor ID), PCI biriminin hangi üretici firma tarafından üretildiğini göstermektedir ve geçerli üretici tanımlayıcı kodları uluslararası "PCI Special Interest Group" tarafından belirlenmektedir.

Durum Bilgisi (Status), PCI yerel yolunda oluşan özel durumları belirtmektedir.

Komut (Command), PCI birimi üzerinde genel kontrol sağlamak amacıyla kullanılmaktadır. Bu alana "0" değerinin yazılması, PCI biriminin PCI yerel yolu ile mantıksal bağlantısının kesilmesini sağlamaktadır. PCI biriminin, bellek erişimi veya giriş_çıkış erişimi ile adreslenebilmesi bu alana yazılan değerler ile belirlenmektedir.

Birim Sınıf Kodu (Class Code), PCI biriminin sınıfını belirtmek için kullanılmaktadır. Bu alanın alabileceği değerler 0 ile 255 arasında değişmektedir. 255 değeri, bu birimin "PCI Special Interest Group" tarafından tanımlanmış sınıfların hiç birine ait olmadığını ve özel bir birim olduğunu belirtmek için kullanılmaktadır.

Üretim Kodu (Revision ID), birim tanımlayıcı kodunun ek bir parçası olarak, PCI biriminin üretim aşamasında geçirdiği değişiklikleri belirtmek amacıyla kullanılmaktadır.

Açılış Test Kodu (Built-in Self Test), PCI biriminin açılış anında durumunu test edebilecek özellikte olup olmadığını, eğer bu test özelliğine sahip ise test sonucunun ne olduğunu belirtmek amacıyla kullanılmaktadır.

Başlık Tipi (Header Type), konfigürasyon alanında 16. adresten başlayan bölümün yapısını tanımlamakta ve PCI biriminin aynı anda birden fazla fonksiyonu yerine getirip getirmediğini belirtmektedir.

Gecikme Süresi (Latency Timer), PCI ana birimleri için, PCI yerel yoluna erişimde geçerli olacak gecikme süresini belirlemektedir.

Önbellek Satır Uzunluğu (CacheLine Size), 32 bit cinsinden, belleğe erişimlerde bir seferde dikkate alınması gereken bilgi miktarını belirtmektedir. Yan belleği etkileyen yazma komutlarında (Memory Write and Invalidate), yazma işlemi sonucunda kaç byte'lık bilginin etkileneceği veya çoklu okuma komutlarında (Read Multiple) bir seferde kaç byte'lık bilginin aktarılacağı, bu alanın değerine bağlı olarak belirlenmektedir.

Taban Adres Saklayıcıları (Base Addresses), PCI birimine erişimlerin hangi adreslerden yapılacağını belirlemek için kullanılmaktadırlar. PCI birimleri en fazla 6 tane taban adres saklayıcı içerebilirler.

Maksimum Gecikme (Max. Latency), PCI birimi tarafından kabul edilebilecek en fazla gecikme süresini belirtmektedir.

Minimum Kabul Süresi (Min. Grant Latency), PCI biriminin, PCI yerel yolu erişim isteği geldiğinde, yerel yolu en az ne kadar zamanda istek yollayan birime bırakabileceğini belirtmektedir.

Kesme Sinyali (Interrupt Pin), PCI biriminin INTA# - INTD# kesme sinyallerinden hangisini kullanacağını göstermektedir.

Kesme Hattı (Interrupt Line), Kesme Sinyali'nin donanım olarak yönlendirilmesini gerçekleştirmektedir.

3. IBM S/370 - S/390 PARALEL KANALI

Bu tez çalışmasında gerçekleştirilen sistem, ana bilgisayar sistemlerindeki paralel kanal yapısının PCI yerel yoluna bağlanmasını ve PCI uyumlu çevre birimlerinin, ana bilgisayar sistemlerine paralel kanal yapısı üzerinden kullandırılmasını sağladığı için, bu bölümde paralel kanal yapısı tanıtılmış, adresleme, seçme ve veri aktarımı özellikleri anlatılmıştır.

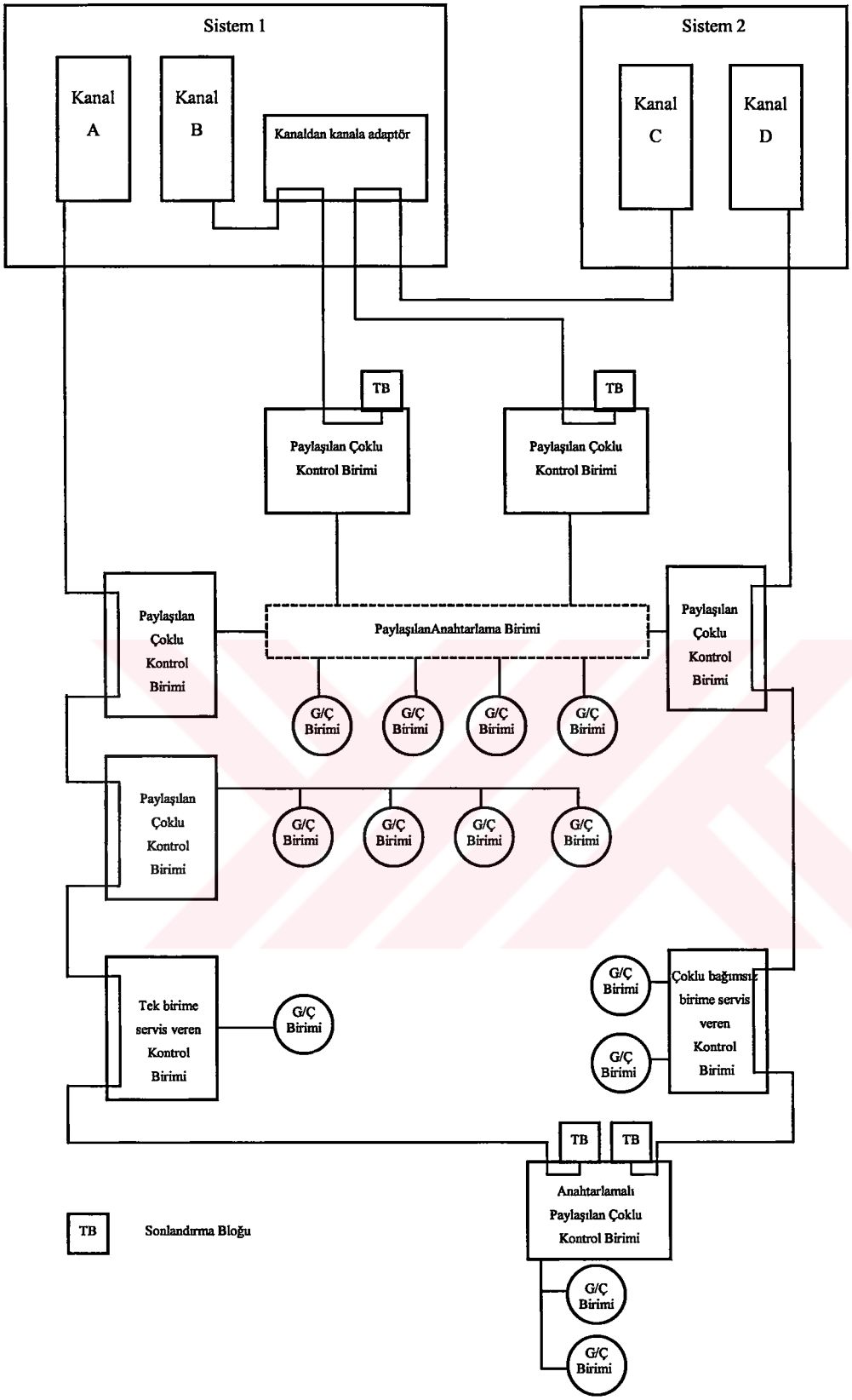
3.1 Paralel Kanalın Yapısı ve Özellikleri

IBM anabilgisayar sistemlerinde giriş_çıkış birimleri, merkezi işlem birimlerine (CPU) kontrol birimleri (control units) ve kanal (channel) yapısı ile bağlıdırlar. Kanal birimleri, kontrol birimleri ile ana bellek arasındaki bilgi akışını sağlamakla görevlidirler. Kanallar, kontrol birimleri ve bunlara bağlı giriş_çıkış birimlerinden oluşan örnek bir sistem konfigürasyonu Şekil 3.1'de verilmiştir.

Giriş_çıkış birimlerinin tasarım ve görevlerinden kaynaklanan fonksiyonel farklılıklarından bağımsız olarak, standart kontrol ve veri akış mekanizmasına sahip kanal yapılarına bağlanabilmeleri ve bu giriş_çıkış birimlerinin yönetilip işletilmesi kontrol birimlerince gerçekleştirilir. Kontrol birimleri, giriş_çıkış birimleri ile aynı fiziksel bütün içerisinde olabilecekleri gibi ayrı birim olarak da tasarlanabilirler.

Kontrol birimleri, kanaldan aldıkları giriş_çıkış komutlarını inceleyip çözdükten sonra, kontrol ettikleri birimin bu komutları yerine getirebilmesi için gerekli sinyalleri üretirler ve komutların işletilmesinden sonra oluşan durum bilgisini (status) de kanala aktarırlar.

Bu yapı içerisinde, her kontrol birimi ile kendisine bağlı olan giriş_çıkış birimi arasında farklı bağlantı ve sinyalleşmeler olmasına rağmen donanım ve yazılım açısından çıkacak zorlukları en aza indirgemek amacıyla kontrol birimleri ile kanal arasında standart bir bağlantı olması öngörülmüştür.



Şekil 3.1 Kanallar, kontrol birimleri ve giriş_çıkış birimlerinden oluşan örnek bir sistem konfigürasyonu (IBM, 1992)

IBM S/370 - S/390 paralel kanal yapısı da öngörülen bu standart bağlantı şekillerinden biridir. Paralel kanal yapısı, aynı kanala bağlanan bir veya birden fazla kontrol birimine giden paralel sinyal hatlarından ve ortak bir sinyalleşme protokolü ile bilgi formatından oluşmaktadır.

Paralel kanalı oluşturan sinyaller kontrol birimlerinin seçilmesi için kullanılanların dışında o kanala bağlı olan tüm kontrol birimlerine giderler. Kontrol birimi seçme işleminde kullanılan sinyaller ise kontrol birimleri arasında seri olarak bir birimden diğerine aktarılmaktadır. Bu sayede seçilmek istenen kontrol birimi bu durumu fark ettiğinde seçme sinyallerinin kendisinden sonra gelen birime aktarılmasına engel olarak kanal ile mantıksal bağlantı kurar. Seçme sinyalleri, kanala tüm kontrol birimleri fiziksel olarak bağlı oldukları halde, sadece bir kontrol biriminin herhangi bir zaman aralığında kanal ile mantıksal bağlantıda olmasını garantiler. Bu bağlantı, kontrol biriminin kendi isteği ile sonlandırılabileceği gibi, kanal tarafından "bağlantı sonlandırılması" durumu oluşturularak da bitirilebilir.

Zamanlama ve elektriksel özellikler nedeniyle bir kanala aynı anda bağlanabilecek kontrol birimi sayısı sekiz ile sınırlandırılmıştır. Bir kanala bağlı kontrol birimlerinin tamamınca aynı anda adreslenebilir giriş_çıkış birimi sayısı da 256 olarak belirlenmiştir. Bu yapılandırmada her kontrol birimi, özelliklerine ve görevine bağlı olarak sadece bir giriş_çıkış birimini kontrol edebildiği gibi aynı özellikte birden fazla giriş_çıkış birimini de kontrol edebilir.

Kanallar ile kontrol birimleri arasındaki bilgi akışı protokolü birden fazla birime paralel erişimi destekleyecek şekilde tasarlanmıştır. Bu protokole göre bir kanal ile ona bağlı olan kontrol birimi arasındaki bilgi akışı tek bir operasyonda tamamlanacak şekilde gerçekleştirilebileceği gibi transfer işlemi birden fazla parçaya bölünerek paralel kanalın zaman paylaşımlı kullanılması ve böylece aynı anda birden fazla kontrol birimi ve giriş_çıkış birimi ile bağlantıda olunması da mümkün olmaktadır.

Paralel kanalı oluşturan sinyallerin yükselmesi (rise) ve düşmesi (fall) karşılıklı kilitlemeli (interlocked) yöntemler ile gerçekleştirildiği için, değişik veri aktarım hızına ve tasarım

özelliklerine sahip kontrol birimleri ve giriş_çıkış birimleri aynı paralel kanala bağlı olarak bir arada çalışabilmektedirler. Çok yüksek hızlı veri aktarım gereksinimi olan disk vb. çevre birimlerinin de paralel kanal üzerinden maksimum performans ile çalışmalarını sağlamak amacıyla sinyalleşmenin kilitlemeli yapılmadığı özel durumlar da tanımlanmıştır. Bu tarz sinyalleşme hem kanal hem de kontrol birimi tasarımı açısından ek tasarım kısıtları ve özellikleri gerektirmektedir.

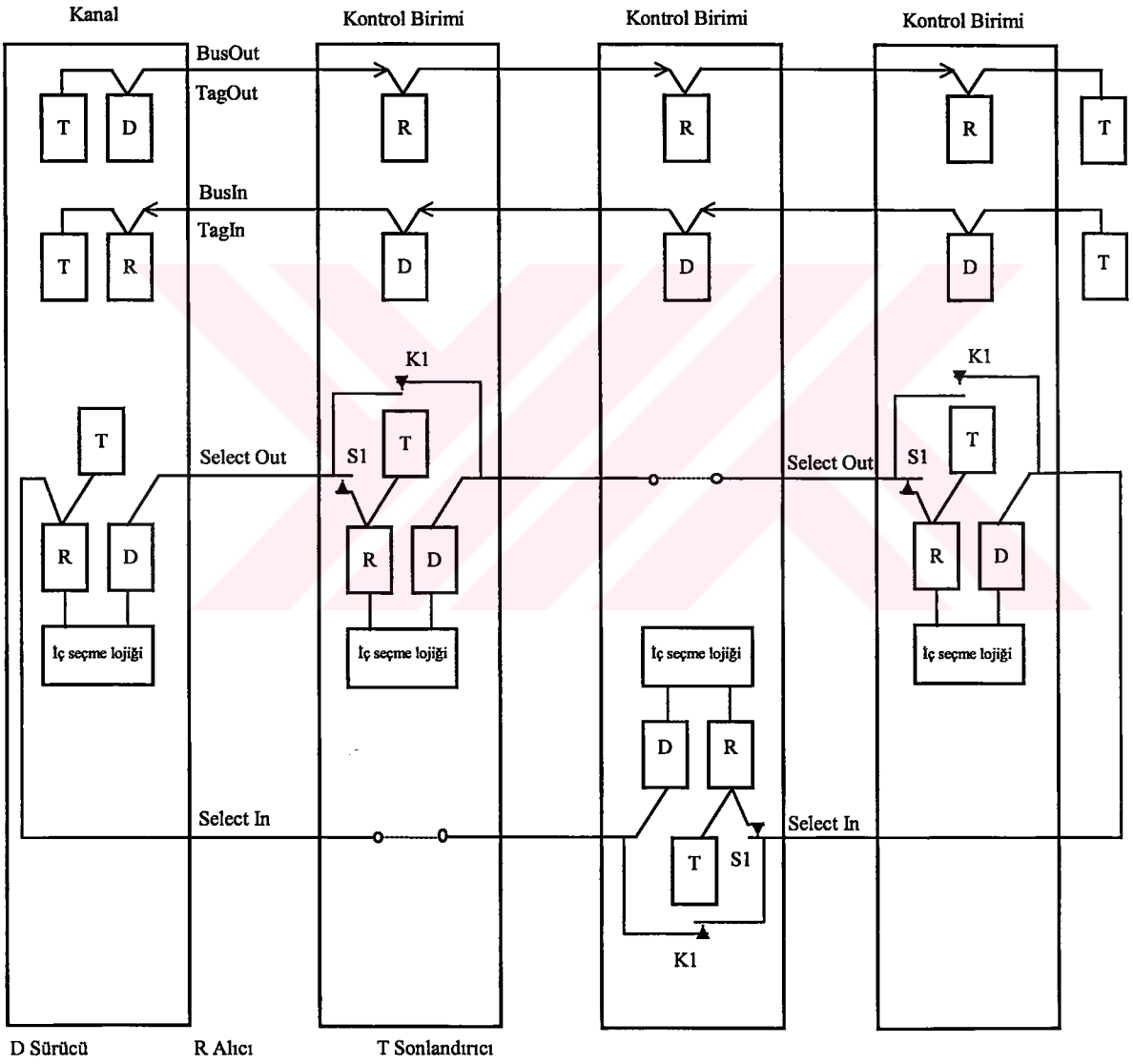
3.2 Paralel Kanalı Oluşturan Sinyaller

Paralel kanal ile kontrol birimleri arasındaki bağlantıyı sağlayan sinyaller, temel olarak dört ana grupta toplanmışlardır. Bu gruplama Tablo 3.1'de verilmiştir.

Tablo 3.1 Paralel kanalı oluşturan sinyal grupları ve görevleri (IBM, 1992)

Hattın adı		Hattın görevi
Bus 0 out eşlik biti Bus 0 out bit 0 Bus 0 out bit 1 Bus 0 out bit 2 Bus 0 out bit 3 Bus 0 out bit 4 Bus 0 out bit 5 Bus 0 out bit 6 Bus 0 out bit 7	Bus 1 out eşlik biti Bus 1 out bit 0 Bus 1 out bit 1 Bus 1 out bit 2 Bus 1 out bit 3 Bus 1 out bit 4 Bus 1 out bit 5 Bus 1 out bit 6 Bus 1 out bit 7	"BusOut" hatları kanaldan kontrol birimlerine bilgi taşımak amacıyla kullanılır. Bu bilgi, kanal ile kontrol birimi arasındaki bağlantının aşamasına bağlı olarak veri, giriş_çıkış birimi adresi, komut veya kontrol bilgisi olabilir. "Bus 1" genişletilmiş yol konfigürasyonunda geçerlidir.
Bus 0 in eşlik biti Bus 0 in bit 0 Bus 0 in bit 1 Bus 0 in bit 2 Bus 0 in bit 3 Bus 0 in bit 4 Bus 0 in bit 5 Bus 0 in bit 6 Bus 0 in bit 7	Bus 1 in eşlik biti Bus 1 in bit 0 Bus 1 in bit 1 Bus 1 in bit 2 Bus 1 in bit 3 Bus 1 in bit 4 Bus 1 in bit 5 Bus 1 in bit 6 Bus 1 in bit 7	"BusIn" hatları kontrol birimlerinden kanala bilgi taşımak amacıyla kullanılır. Bu bilgi, kontrol birimi ile kanal arasındaki bağlantının aşamasına bağlı olarak veri, seçilen giriş_çıkış birimi adresi, durum bilgisi (status information), algılama bilgisi (sense information) olabilir. "Bus 1" genişletilmiş yol konfigürasyonunda geçerlidir.
Address Out Command Out Service Out Data Out	Address In Status In Service In Data In	Etiket (tag) hatları karşılıklı kilitleme (interlocking) ve kontrol sinyalleşmesi için kullanılır. Ayrıca özel sekanslar da bu hatlar ile ifade edilir.
Operational Out Operational In Hold Out Select Out Select In Suppress Out Request In Disconnect In		Kontrol birimi seçme hatları, giriş_çıkış birimlerinin tespit edilmesi ve seçilmesi için kullanılır. "Disconnect In" giriş_çıkış hatası alarmı özelliği olan kontrol birimlerinde mevcuttur.

Paralel kanal ile kontrol birimleri arasındaki veri alışverişi BusIn ve BusOut olarak isimlendirilen yolları oluşturan sinyaller ile gerçekleştirilmektedir. Bu sinyallerin ne şekilde değerlendirilmesi ve buna bağlı olarak nasıl bir işlem gerçekleştirilmesi gerektiği ise TagIn ve TagOut yollarındaki sinyaller ile belirlenmektedir. Sadece kanala en yakın kontrol birimi, kanal ile doğrudan bağlantı halindedir, diğer kontrol birimleri kendilerinden bir önce gelen kontrol birimine bağlanırlar. Bir kanal ve üç adet kontrol biriminden oluşan bir yapı Şekil 3.2'de görülmektedir.



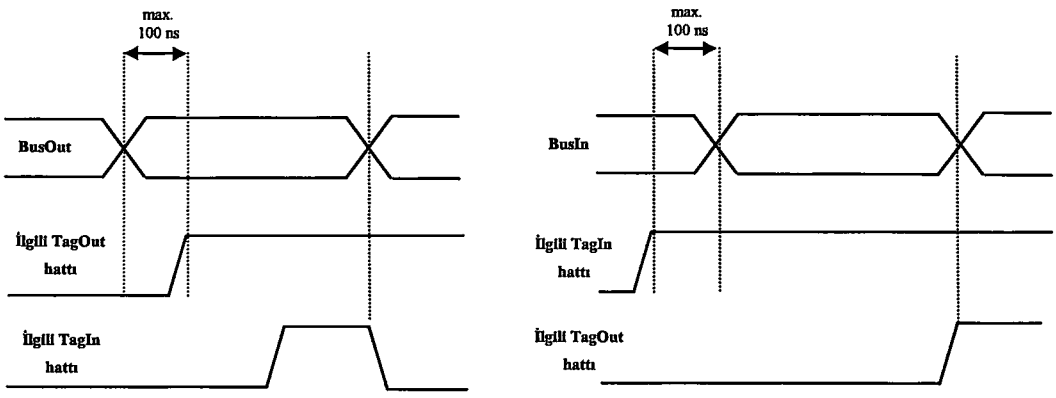
Şekil 3.2 Bir kanal ve 3 kontrol birimi arasındaki bağlantı (IBM, 1992)

Bu yollar, isimlerindeki "In" ve "Out" kısımları paralel kanal referans alınacak şekilde belirlenmiştir. Buna göre BusOut yolu kanaldan kontrol birimine veri taşındığını

belirtmekte, TagIn ise kontrol biriminden kanala kontrol bilgisi aktarıldığını göstermektedir.

BusOut ve BusIn yolları sekizer adet veri hattından ve bir adet tek eşlik hattından oluşmaktadır. BusOut yolu, kanal ile kontrol birimi arasındaki haberleşmenin bulunduğu aşamaya bağlı olarak, kanaldan kontrol birimine adres, komut ve veri aktarımı için kullanılmaktadır. BusIn yolu ise kontrol biriminden kanala adres, durum bilgisi ve veri aktarımı sağlamaktadır.

BusIn ve BusOut yollarındaki bilgilerin nasıl değerlendirileceği ve ne kadar süreyle geçerli oldukları TagIn ve TagOut yollarındaki sinyallerin aktif veya pasif olmaları ile belirlenmektedir. Kullanılan lojik elemanlardan ve kanal ile kontrol birimi arasındaki kablodan kaynaklanabilecek sinyal gecikmelerini dengelemek ve sinyallerin kararlı hale gelmelerini sağlamak amacıyla kanal, BusOut yolundaki bilgileri, bu bilgileri tanımlayan TagOut yolundaki hattı aktif hale getirmeden en geç 100 ns önce aktif hale getirir. BusOut yolundaki bilgiler, bu bilgilerin kontrol birimi tarafından kabul edildiğini gösteren TagIn hattının aktif durumdan pasif duruma dönmesine kadar geçerliliklerini korurlar. Aynı şekilde kontrol biriminden kanala gelen bilgilerde de bilgiyi tanımlayan TagIn hattının aktif hale gelmesinden en geç 100 ns sonra BusIn yoluna geçerli veri çıkarılır. BusIn yolundaki bilgi, kanal tarafından kabul edildiğini gösteren TagOut hattı aktif olana kadar geçerliliğini korur (Şekil 3.3).



Şekil 3.3 Kanal ile kontrol birimi arasındaki bilgi aktarımında uyulması gerekli zamanlama

3.3 Kontrol Birimi Seçme Kontrolleri ve Etiket Sinyalleri

Kanal ile kontrol birimlerinin haberleşmesi sırasında, kontrol birimlerinin seçilmesini ve bu sayede giriş_çıkış birimlerine erişilmesini sağlayan sinyaller aşağıda açıklanmıştır.

3.3.1 Operational Out sinyali

Operational Out sinyali, kanaldan tüm kontrol birimlerine gider ve kanalın servis verebilir durumda olup olmadığını belirtir. Suppress Out sinyali dışında kanaldan kontrol birimlerine giden tüm sinyaller, Operational Out sinyali aktif iken geçerlidirler. Operational Out sinyalinin geri çekilmesi, kanal ile kontrol birimi arasındaki işlemlerin sıfırlanması gerektiğini belirttiği gibi, kontrol birimlerinin o an için aktif olan hatlarını da en geç 1.5 ms içerisinde geri çekmeleri gerektiğini belirtir.

3.3.2 Request In sinyali

Request In sinyali ile kontrol birimleri kanala, servis isteğinde bulunduklarını belirtirler. Servis isteğinin gelmesi üzerine kanal, kontrol birimlerinin servis isteğini karşılamak amacıyla bir seçme işlemi başlatır. Request In sinyali aynı anda birden fazla kontrol birimi tarafından aktif hale getirilebilecek yapıdadır. Request In sinyali bastırılabilir (suppressible) bir durum bilgisi sunmak amacıyla aktif hale getirildiğinde, kanal tarafından Suppres Out sinyali gönderilirse, ilgili kontrol birimi Request In sinyalini geri çeker.

3.3.3 Address Out sinyali

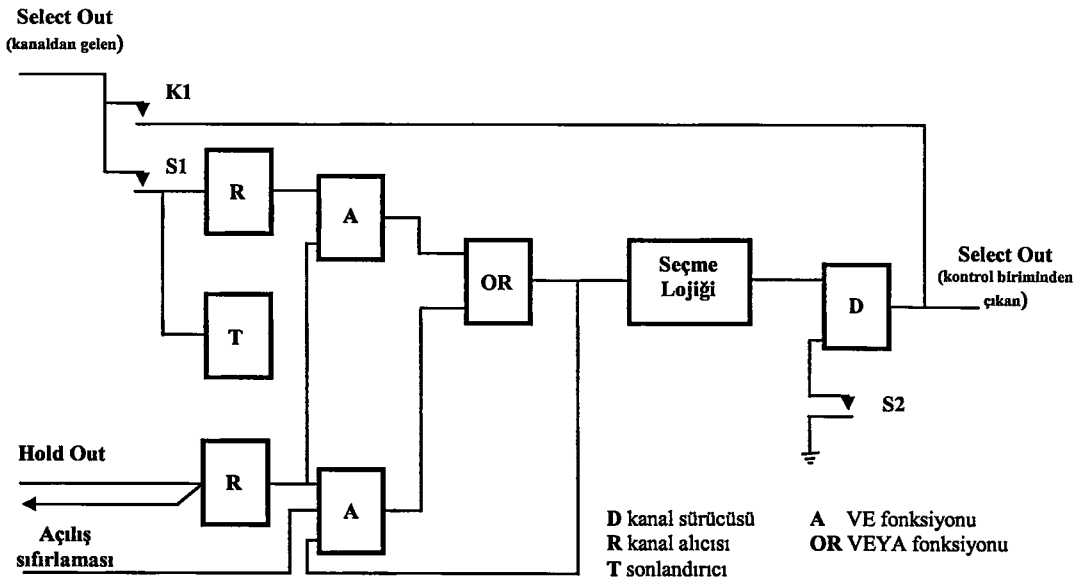
Kanal, Address Out sinyali ile kendisine bağlı tüm kontrol birimlerine, BusOut yoluna erişilmek istenen giriş_çıkış birimine ait adresin çıkarıldığını belirtir. Adresi belirtilen giriş_çıkış cihazını kontrol etmekle görevli kontrol birimi, Address Out sinyali aktif iken Select Out sinyali de aktif olursa Operational In sinyalini aktif hale getirerek seçme işlemini tamamlar. Address Out sinyali, ancak Select In, Status In ve Operational In

sinyalleri aktif değilken aktif hale getirilebilir ve Operational In veya Select In aktif olana kadar aktif durumda kalır.

3.3.4 Select Out, Hold Out ve Select In sinyali

Kontrol birimlerinin seçilmesi işlemi, Select Out, Select In ve Hold Out sinyalleri ile gerçekleşir. Select Out ve Select In sinyalleri kanaldan başlayıp, kanala bağlı tüm kontrol birimlerini dolaşan ve en sonda yer alan kontrol biriminden sonlandırma bloğu ile kanala geri dönen bir çevrim oluştururlar. Bu sayede kontrol birimleri, seçme işlemine yapılarına bağlı olarak Select Out veya Select In sinyali ile katılabilirler. Select Out sinyali için geçerli olan kuralların tamamı, Select In sinyalinin Select Out yerine kullanılması durumunda Select In için de geçerlidir. Seçme işleminde Select In sinyalini kullanan kontrol birimi, kanal tarafından başlatılan seçme işleminin en son kontrol birimine kadar ulaşmasını ve geri dönüş yolu üzerinde kendisine gelmesini bekleyeceği için daha düşük önceliğe sahip olacaktır. Kontrol birimleri adresleme ve seçme işleminin hedefinin kendileri olmadığını anladıkları zaman Select Out veya Select In sinyallerini kendilerinden sonra gelen kontrol birimine aktarırlar. Seçme işleminin hedefi olmaları durumunda kontrol birimleri bu sinyallerin kendilerinden sonra gelen birime devam etmesini engellerler. Bu sebeple, kontrol birimleri özellikle açılış ve kapanış anlarında Select Out veya Select In sinyallerinde elektriksel bir kesinti oluşmaması için özel önlemler alırlar. Şekil 3.4'de bu amaçla kullanılabilecek örnek bir devrenin yapısı verilmiştir. Select Out sinyalinin ilerlemesini engelleyen kontrol birimi, seçme işlemini tamamlamak amacıyla Operational In sinyalini aktif hale getirir. Sinyali bir sonraki kontrol birimine devam ettiren kontrol birimleri kanal tarafından seçilmek için bir sonraki seçme çevrimini beklemek zorundadırlar.

Hold Out sinyali, kontrol birimi seçilmesi işleminde Select Out sinyali ile beraber kullanılır. Bu sinyal aynı anda tüm kontrol birimlerine gittiği için Select Out sinyalinin, kontrol birimleri arasındaki seri aktarım sebebiyle, geri çekilmesi sırasında oluşan gecikmeyi engeller ve aynı zamanda seçme işlemini senkronize eder. Hold Out sinyali aktif değilken, kontrol birimleri çıkışlarındaki Select Out sinyalini aktif olmayan konumda tutarlar.



Şekil 3.4 Select Out sinyalinin sürekliliğini sağlamak için örnek devre (IBM, 1992)

3.3.5 Operational In sinyali

Operational In sinyali, kanala kontrol birimi seçme işleminin başarıyla tamamlandığını belirtmek amacıyla kullanılır. Seçilmiş olan giriş_çıkış biriminin adresi, BusIn yolu üzerinden Address In sinyali ile beraber kanala yollanır. Operational In sinyalinin aktif olabilmesi için, kontrol biriminin Adress Out zamanında BusOut üzerinden adresi yollanan giriş_çıkış birimini kontrol ediyor olması ve kontrol birimi girişindeki Select Out sinyalinin aktif, çıkışındaki Select Out sinyalinin ise pasif konumda olması gereklidir.

Operational In sinyali, kanal ile veri aktarımı devam ettiği ve Select Out sinyali aktif olduğu sürece aktif konumunu korur. Operational In sinyali aktif iken Select Out sinyali geri çekilirse Operational In sinyali aktarılacak veri sayısına bağlı olarak ya hemen ya da son veri aktarıldıktan sonra geri çekilmek zorundadır.

3.3.6 Address In sinyali

Kontrol birimleri, seçilmiş olan giriş_çıkış birimine ait adresin BusIn yolu üzerinde olduğunu kanala Address In sinyali ile belirtirler. Kanal Address In sinyaline cevap olarak Command Out sinyalini aktif hale getirir. Command Out sinyalinin kanal tarafından geri çekilebilmesi için, kontrol biriminin Address In sinyalini geri çekmesi gerekmektedir.

3.3.7 Command Out sinyali

Command Out sinyali, kanal tarafından giriş_çıkış birimlerine, Address In, Status In, Data In veya Service In sinyallerinin aktif olması durumunda cevap vermek amacıyla kullanılır. Command Out sinyalinin aktif olması, BusIn yolu üzerindeki bilgiye artık ihtiyaç duyulmadığını ve bu bilginin değişebileceğini belirtir. Command Out sinyali Data In veya Service In sinyallerine cevap vermek amacıyla kullanılırsa, gerçekleştirilmekte olan veri aktarımının sonlandırılması gerektiğini gösterir. Status In sinyaline karşılık Command Out sinyali kullanıldığında, kanalın o an için yüklü olduğu ve kontrol biriminin kanala aktarmak istediği durum bilgisini saklaması gerektiği belirtilir. Kanal kendi isteği ile kontrol birimini seçti ise Address In sinyaline Command Out ile cevap vermesi BusOut yolu üzerinde giriş_çıkış birimince işlenmesini istediği komutu yolladığını gösterir. Kontrol birimi Request In ile seçilmesini sağladı ise, Address In sinyaline cevap olarak verilen Command Out sinyali, kanalın kontrol birimi ile haberleşmeye hazır olduğunu belirtir.

3.3.8 Status In sinyali

Status In sinyali, seçilmiş olan kontrol biriminin durum bilgisini BusIn yoluna çıkardığını gösterir. Kanal bu sinyale cevap olarak ya Service Out ya da Command Out sinyalini aktif hale getirir. Status In sinyali kanala, kontrol birimlerinde oluşabilecek kısa süreli meşgul olma durumunu aktarmak amacıyla da kullanılır. Bu durumda kontrol birimi Select Out sinyaline cevap olarak Status In sinyalini aktif hale getirir.

3.3.9 Service Out sinyali

Service Out sinyali, seçilmiş olan giriş_çıkış birimine Service In veya Status In sinyallerinin aktif olduğu durumlarda olumlu cevap vermek için kullanılır. Service Out sinyalinin aktif hale gelmesi, gerçekleştirilen giriş_çıkış işleminin yönüne göre ya BusIn üzerindeki verinin kanal tarafından kabul edildiğini ya da kanalın istenen veriyi BusOut yoluna çıkardığını belirtir.

3.3.10 Service In sinyali

Kontrol birimleri bu sinyal aracılığı ile kanala, seçilmiş olan giriş_çıkış biriminin veri aktarımı yapmaya hazır olduğunu bildirirler. Kanal, bu sinyale Service Out veya Command Out sinyali ile cevap verir.

Servis In sinyali, okuma komutlarında, kanalın ihtiyaç duyduğu verinin BusIn yoluna çıkarıldığını, yazma komutlarında ise kanaldan BusOut yolu üzerinden yeni veri beklendiğini belirtir. Kanalın özellikle disk veya teyp türü giriş_çıkış birimleri ile çalışılırken Service In sinyaline belli bir süre içerisinde cevap verememesi halinde kontrol birimleri hata durumu oluştururlar.

3.3.11 Suppress Out sinyali

Bu sinyal, diğer TagOut sinyallerden farklı olarak herhangi bir anda diğer sinyallerin seviyelerinden bağımsız olarak aktif hale gelebilir. Tek başına veya diğer TagOut sinyalleri ile beraber kullanılarak veri aktarımının geçici bir süre durdurulmasını veya kontrol birimlerinin sıfırlanıp başlangıç konumuna getirilmesini sağlar.

3.4 Paralel Kanalda Temel Arabirim İşlemleri

Bir kanal ile kontrol birimleri arasındaki haberleşme belli adımlar izlenerek gerçekleştirilir. Buna göre, öncelikle kanal ile kontrol birimi arasında bir seçme işleminin gerçekleştirilmesi, daha sonra veri aktarımı aşamasına geçilmesi ve sonuç olarak da haberleşmenin düzgün olarak sonlandırılması gerekmektedir. Bu aşamaların işleyişi aşağıdaki alt başlıklarda incelenmiştir.

3.4.1 Kontrol birimlerinin seçilmesi

Kanal ile kontrol birimleri arasındaki seçme işlemi, kanal tarafından başlatılabileceği gibi kontrol birimi kanala seçme işlemini başlatması için istekte de bulunabilir.

3.4.1.1 Kanal tarafından başlatılan seçme işlemi

Kanal tarafından erişilmek istenen giriş_çıkış biriminin adresi BusOut yoluna çıkarılır ve Address Out sinyali aktif hale getirilir. Bu durumu fark eden kontrol birimleri BusOut üzerinde yer alan adresin kendi kontrollerindeki bir giriş_çıkış birimi olup olmadığını kontrol ederler. Bu adresin geçerli olabilmesi için adrese ait eşlik bitinin doğru olması gereklidir.

Kanal, daha sonra Select Out sinyalini aktif hale getirerek seçme işleminin ikinci aşamasına geçilmesini sağlar. Adreslenen giriş_çıkış biriminin bulunduğu kontrol birimi Select Out sinyalinin aktif olduğunu fark ettiği anda bu sinyalin kendisinden sonra gelen kontrol birimine aktarılmasını engeller ve Operational In sinyalini aktif hale getirir. Kanal, kontrol biriminin işleme devam edebilmesini sağlamak amacıyla Address Out sinyalini geri çeker. Kontrol birimi adreslenen giriş_çıkış biriminin adresini BusIn yoluna çıkarır ve Adress In sinyalini aktif hale getirir. Kanal, giriş_çıkış biriminin adresini kontrol ettikten sonra bu birim tarafından işletilmesini istediği komutu BusOut yoluna çıkarır ve Command Out sinyalini aktif hale getirir. Adreslenen birim bu komutu inceler ve hatasız olduğunu belirlerse Address In sinyalini aktif hale getirerek komutu kabul ettiğini belirtir ve Status In sinyalini aktif hale getirerek bu komuta ait ilk durum bilgisini kanala sunar. Bu ilk

durum bilgisi kanala, kontrol biriminin komutu kabul edip etmediğini belirtir. Komutun kabul edildiği durumlarda kanal, giriş_çıkış işleminin başarı ile başlatıldığını anlar ve Status In sinyaline cevap olarak Service Out sinyalini aktif hale getirir. İlk durum bilgisininin kabul edilmediği durumlarda Command Out sinyali aktif hale getirilerek kontrol birimine bildirilir.

3.4.1.2 Kontrol biriminin kısa süreli meşgul olma durumu

Kanal tarafından başlatılan giriş_çıkış birimi seçme işlemi sırasında kontrol birimi, kanala cevap vermesini ve seçme işlemini tamamlamasını engelleyecek bir giriş_çıkış işlemi gerçekleştiriyorsa bu durumu kısa süreli meşgul olma ile bildirir. Seçme işlemi, kontrol biriminin Operational In sinyalini aktif hale getirmesini gerektiren aşamaya kadar ilerler ve bu aşamada kontrol birimi, meşguliyetini belirten durum bilgisini BusIn yoluna çıkararak Status In sinyalini aktif hale getirir. Bu durumda kanal, durum bilgisini okuduğunu belirtmek ve seçme işlemini sonlandırmak için Select Out sinyalini geri çeker.

3.4.1.3 Kontrol birimi tarafından başlatılan seçme işlemi

Kontrol birimleri, kanal ile haberleşmek istediklerini Request In sinyalini aktif hale getirerek belirtirler. Request In sinyalinin aktif olması ile kanal, mümkün olan ilk anda yeni bir seçme işlemi başlatır. Bu seçme işleminde BusOut yoluna giriş_çıkış biriminin adresinin çıkarılması ve Address Out sinyalinin aktif hale getirilmesi söz konusu değildir. Kanal, doğrudan Select Out sinyalini aktif hale getirir. Bu durumu fark eden kontrol birimi, kanal tarafından başlatılan seçme işleminde olduğu gibi seçme işlemine devam eder.

3.4.2 Veri aktarımı

Veri aktarımı aşamasına ancak başarılı bir seçme işleminden sonra geçilebilir. Veri aktarımının hangi yönden yapılacağı, seçme işleminde kontrol birimine aktarılmış olan komuta göre belirlenir. Kontrol biriminden kanala veri aktarılacağı zaman, kontrol birimi

BusIn yoluna veriyi çıkarır ve Service In sinyalini aktif hale getirir. Kanal bu veriyi okuduğunu Service Out sinyalini aktif hale getirerek belirtir. Bu sayede Service In ve Service Out sinyalleri karşılıklı el sıkışmalı yapı ile veri aktarımı yapılmasını sağlarlar. Service In sinyali, Service Out sinyali aktif olana kadar aktif halde tutulmalıdır ve Service Out sinyali de ancak Service In sinyali geri çekildiğinde geri çekilmelidir.

Kontrol biriminin kanaldan veri beklediği durumda ise kontrol birimi Service In sinyalini aktif hale getirir ve kanalın BusOut yoluna veri çıkarmasını ve Service Out sinyalini aktif hale getirmesini bekler. Bu durumda da Service In ve Service Out sinyalleri arasındaki ilişki kanala veri aktarılmasında olduğu gibidir.

3.4.3 Giriş_çıkış işleminin sonuçlandırılması

Giriş_çıkış işleminin sonlandırılması kanal veya kontrol birimi tarafından başlatılabilir. Kontrol birimi tarafından başlatılması, giriş_çıkış işleminin sonuçlandığını ve kontrol biriminin bu işleme ait sonucu kanala aktarmak istediğini gösterir. Bu durumda kanala sadece bir adet sonuç durumu aktarılır. Giriş_çıkış işleminin sonuçlandırılması, kanal tarafından başlatıldığı zaman, giriş_çıkış birimin yürütmekte olduğu işlemin durumuna göre kanala birden fazla sonuç durumu aktarılabilir. Giriş_çıkış işleminin sonuçlanmasını gerektiren üç durum söz konusudur:

1. Kanal, işlemin sonuçlandığını fark etmiştir. Bu durumda kontrol birimi kendisinden Service In ile servis istediğinde Command Out işlemin sonuçlandırılmasını ister.

2. Kanal ve kontrol birimi aynı anda işlemin sonuçlandığını fark etmişlerdir.

3. Kontrol birimi, kanaldan önce işlemin sonuçlandığını fark etmiştir.

3.5 Paralel Kanalda Adresleme

Paralel kanala bağlanan her giriş_çıkış birimine 8 bit'ten oluşan bir adres atanmıştır. Kanal üzerinde işlemlerin doğru yürütülebilmesi için aynı adresin birden fazla giriş_çıkış birimine atanmamış olması gereklidir. Bir giriş_çıkış birimi bağlı olduğu kontrol birimini başka giriş_çıkış birimleri ile paylaşmıyorsa, bu birim için 0 ile 255 arasındaki herhangi bir adres kullanılabilir. Aynı kontrol biriminin birden fazla giriş_çıkış birimini kontrol etmesi durumunda, giriş_çıkış birimlerine adresler, 16'yı geçmeyecek şekilde gruplar halinde verilmelidir. 16'dan daha fazla giriş_çıkış birimi kontrol edilmesi gerektiğinde bu birimler birbirini takip etme zorunluluğu olmayan gruplara ayrılabilirler. Grubun başlangıç adresi aynı zamanda kontrol birimine atanamış olan adresi ifade eder. Örneğin, bir kontrol biriminde 12 adet giriş_çıkış birimi kontrol ediliyorsa ve bu birimlere onaltılık sistemde 20 ile 2b arasında adresler atanmışsa 20 adresi hem kontrol biriminin adresini hem de birinci giriş_çıkış biriminin adresini göstermektedir.

3.6 Paralel Kanal Komutları

Paralel kanal yapısı, giriş_çıkış birimlerinin yapılarından ve özelliklerinden kaynaklanan farklılıkları, kullanıcı programlarına yansıtmayacak şekilde giriş_çıkış komutları tanımlamıştır. Giriş_çıkış komutları bir byte uzunluğundadır ve byte içerisinde sağ taraftaki bitler komut türünü, sol taraftaki bitler ise aynı tür içerisindeki farklı durumları belirtirler. Sol taraftaki bitlerin değerlendirilmesi, komutun işletileceği giriş_çıkış birimine aittir. Paralel kanalda tanımlı komutlar ve bu komutların iç yapıları Tablo 3.2'de verilmiştir. Bir komutun kontrol birimi tarafından kabul edilmesi için öncelikle eşlik bitinin doğru olması gerekir. Read, read backward, write, control, sense ve test komutları temel komut kümesini oluştururlar. Kontrol birimi, komutun kendisine ve bu komutun işletileceği giriş_çıkış biriminin özelliğine bağlı olarak kanaldan komuta ait diğer parametrelerin de aktarılmasını isteyebilir. Bunu gerçekleştirmek için kontrol birimi, ya ilk durum bilgisini kanala sunduktan sonra kanal ile bağlı kalmaya devam eder ve kanaldan parametreleri alır ya da daha sonra kanala bir servis isteği yollayıp kanalın seçme işlemi başlatmasını sağlar.

Tablo 3.2 Paralel kanalda tanımlı komutlar ve bu komutların yapıları (IBM, 1992)

Paralel Kanal Komutları	Komutları oluşturan bitler								
	P	0	1	2	3	4	5	6	7
Test	P	M	M	M	M	0	0	0	0
Test I/O	1	0	0	0	0	0	0	0	0
		0	0	0	1				
Tanımsız / saklı tutulmuş	P	-	-	-	-	0	0	0	0
		1	1	1	1				
Sense	P	M	M	M	M	0	1	0	0
Basic sense	0	0	0	0	0	0	1	0	0
Sense ID	1	1	1	1	0	0	1	0	0
Tanımsız / saklı tutulmuş	P	M	M	M	M	1	0	0	0
Read backward	P	M	M	M	M	1	1	0	0
Write	P	M	M	M	M	M	M	0	1
Read	P	M	M	M	M	M	M	1	0
Basic read	0	0	0	0	0	0	0	1	0
Control	P	M	M	M	M	M	M	1	1
No-operation	1	0	0	0	0	0	0	1	1

Read komutu, giriş_çıkış birimlerinden kanala veri aktarılmasını başlatır. Veriler kanala, yazma işleminde izlenen sıra ile yollanırlar. Read komutunun ozel bir durumu olan **Basic read** komutu, adreslenen giriş_çıkış biriminden işletim sisteminin yüklenebilmesini sağlar.

Read backward komutu ise verilerin kanala yazma işleminde izlenen sıranın tersi ile yollanmasını sağlar.

Write komutu, kanaldan giriş_çıkış birimlerine veri aktarılmasını sağlar. Zamanlama ve sinyallerin akışı, Read komutunda olduğu gibidir.

Control komutu Write komutu ile aynı yapıdadır ve kontrol birimine giriş_çıkış birimi veya kanal bağlantısı ile ilgili yapılması istenen kontrol türündeki işlemleri belirtir. Kontrol birimleri, Control komutunda yer alan M bitlerini değerlendirerek yapılması istenen işlemi çözerler. Bu komutlar, giriş_çıkış birimlerinin ikinci seviyeden adreslenmesi, kanal ile

kontrol birimi arasında senkronizasyon sağlanması gibi çok değişik işlemleri yerine getirirler.

Sense komutu, giriş_çıkış birimlerinin fiziksel durumları hakkında bilgi içeren iç saklayıcıların değerlerini okumak için kullanılır. Her giriş_çıkış biriminde bu tür iç saklayıcılardan en çok 32 tane olabilir. **Basic sense** komutu geldiğinde kontrol birimi bu iç saklayıcıların değerlerini kanala aktarır. Bu komut hem giriş_çıkış birimlerinin istenilen komutları yerine getirip getiremeyeceğinden emin olmak için hem de giriş_çıkış hataları sonrasında birimlerde oluşan donanımla ilgili hataları belirlemek için kullanılır. Sense komutunun özel bir durumu olan **Sense ID** komutu, giriş_çıkış biriminin tipini, modelini ve bağlı olduğu kontrol biriminin tipini ve modelini belirlemek için kullanılır. Sense ID komutunun döndürdüğü bilgiler Tablo 3.1de gösterilmiştir.

Tablo 3.3 Sense ID komutunun sonucu (IBM, 1992)

Sense ID komutunun döndürdüğü byte	İçeriği
0	FF (onaltılık sistem)
1-2	Kontrol biriminin tipini belirleyen numara
3	Kontrol biriminin model numarası
4-5	Giriş_çıkış biriminin tipini belirleyen numara
6	Giriş_çıkış biriminin model numarası

Test I/O komutu, adreslenen giriş_çıkış biriminden, daha önce yürütülmüş fakat sonucu henüz kanala aktarılmamış giriş_çıkış işlemlerine ait durum bilgisini almak amacıyla kullanılır. Adreslenen giriş_çıkış biriminde kanala aktarılmamış durum bilgisi bulunmuyorsa Test I/O komutuna cevap olarak 0 değeri döndürülür. Test I/O komutu işletildiği giriş_çıkış biriminde herhangi bir giriş_çıkış işlemi başlatmaz.

3.7 Paralel Kanalda Yüksek Hızlı Veri Aktarımı

Paralel kanal üzerinde veri aktarımının, normal durumlarda Service In ve Service Out sinyallerinin karşılıklı el sıkışması ile gerçekleştirilmesi, yüksek hızlı veri aktarımlarını engellemektedir. Bu yöntemle erişilebilen hızlar, yüksek hızlı veri aktarımı özelliğine sahip çevre birimlerinin ihtiyacını karşılıyamamaktadır. Bu tür çevre birimlerinin veri aktarım ihtiyaçlarını karşılayabilmek amacıyla Data In ve Data Out olmak üzere iki sinyalden daha yararlanılır. Data In sinyali, Service In sinyali ile Data Out sinyali ise Service Out sinyali ile dönüşümlü olarak kullanılır.

Kanal ile kontrol birimi arasındaki mesafe kısa olduğunda, bir birimden yollanan sinyal, yayılım gecikmesinden etkilenmeden diğerine erişeceği için, ardışık sinyal aktarımında, her sinyal kendisinden önce gönderilen sinyalin cevabının gelmesini beklemek zorundadır. Aradaki mesafenin, sinyallerin yayılım gecikmesinden etkilenecek şekilde, arttırılması ile birimler yolladıkları sinyalin cevabının belli bir gecikme ile geleceğini bildikleri için, cevap bekleme zorunluluğu olmaksızın yeni sinyal yollayabilirler. Bu sebeple, daha yüksek aktarım hızlarına ulaşılabilmesi amacıyla kontrol birimleri ile kanal arasındaki mesafenin de arttırılması gereklidir.

Data In sinyali yüksek hızlı veri aktarım özelliğine sahip kontrol birimleri tarafından, veri aktarımı sırasında Service In sinyaline alternatif olarak kullanılır. Kontrol biriminden kanala veri aktarılacağı zaman kontrol birimi BusIn yoluna veriyi çıkardıktan sonra dönüşüm sırasına göre ya Service In ya da Data In sinyalini aktif hale getirir. Kanaldan kontrol birimine veri aktarımı durumunda ise kontrol birimi, Data In veya Service In sinyalini aktif hale getirerek BusOut yolunda yeni veri beklediğini belirtir. Kanal, Data In sinyali aktif hale geldiğinde Data Out veya Command Out sinyali ile cevap verir. Sinyallerin dönüşümlü olarak kullanılması sonucunda belli anlarda kontrol biriminde bir süre için hem Service In hem de Data In sinyalleri aktif durumda olabilirler. O an için aktif olma sırası kendisine gelmiş olan sinyal, diğer sinyal geri çekilene kadar kanal tarafından geçerli kabul edilmez.

Yüksek hızlı veri aktarım özelliğine sahip olan kanalların da Data Out sinyaline sahip olması gereklidir. Bu sinyal kanal tarafından Data In sinyalini cevaplamak amacıyla kullanılır. Data Out sinyalinin aktif hale gelmesi, okuma türü komutlarda, BusIn yolundaki bilginin kabul edildiğini, yazma türü komutlarda ise BusOut yoluna gerekli verinin çıkarıldığını belirtir.

3.8 Paralel Kanalda Veri Akışı (Data Streaming) Yöntemi ile Veri Aktarımı

Veri akışı yöntemi, paralel kanala bağlı giriş_çıkış cihazları ile kanal arasında saniyede 4.5 Mbyte'a varan hızlarda veri aktarımı yapılmasını sağlar ve kontrol birimleri ile kanal arasında 122 metre mesafe olmasına izin verir. Bu aktarım yönteminin en önemli özelliği aktarım esnasında el sıkışma ve kilitleme yöntemleri kullanmaması ve hem kontrol birimlerinde hem de kanallarda çok hassas zamanlamaların uygulanmasını gerektirmesidir. Veri akışı yöntemini uygulayan kanallar ve kontrol birimleri için 3 Mbyte/s ve 4.5 Mbyte/s olmak üzere iki adet standart aktarım hızı tanımlanmıştır. 4.5 Mbyte/s aktarım hızı, uygulandığı birimlerde tasarım ve zamanlama kurallarına özellikle uyulmasını gerektirir. Paralel kanalı oluşturan sinyallerde, hangi veri akışı hızınının kullanılacağını belirten bir sinyal olmadığı için, çalışma sırasında bu iki hız arasında geçişlere izin verilmez. Veri akışı yöntemini uygulayan kanallar, standart veri aktarım yöntemlerini de uygularlar. Her veri aktarım işleminin başında kontrol birimleri, kanal ile hangi yöntemi kullanarak veri aktarmak istediklerini belirtirler. Veri aktarım işleminin, Data In sinyali aktif hale getirilerek başlatılması, veri akışı yöntemi ile haberleşileceğini gösterir. Aynı işlemin Service In sinyali aktif hale getirilerek başlatılması, veri aktarımının ya karşılıklı kilitlemeli yöntemle ya da yüksek hızlı veri aktarımı yöntemi ile yapılmak istendiğini belirtir. Veri akışı yönteminde aktarım hızı, tanımlı olan 3 Mbyte/s veya 4.5 Mbyte/s'lik üst sınırı aşamaz. Herhangi bir andaki gerçek aktarım hızı, Service In veya Data In sinyalinin periyoduna bağlıdır.

Karşılıklı kilitleme ve el sıkışmanın uygulandığı aktarım yöntemlerinden farklı olarak, veri akışı yönteminde Service In ve Data In sinyallerinin aktif hale getirilmeleri ve geri çekilmeleri, Service Out ve Data Out sinyallerinin durumlarından bağımsızdır. Kontrol birimleri, uygulanan aktarım hızına uygun periyod ile Service In ve Data In sinyallerini

dönüşümlü olarak aktif hale getirirler. Kanal ise bu sinyallerin aktif hale geldiğinde cevap olarak Service Out veya Data Out sinyallerini aktif hale getirir. Kanal, kontrol birimi ile arasında uyumu sağlayabilmek için, Service Out veya Data Out sinyallerini ancak bunlara denk düşen kontrol birimi sinyalleri geri çekildiğinde geri çeker.

Veri akışı yönteminin en önemli özelliği, alıcı ve verici taraflar arasındaki zamanlama bağımlılığını en aza indirip bağlantı üzerinde bir iş hattı oluşturmasıdır. Bu özellik sayesinde kontrol birimini kanala bağlayan BusIn yolunun hatları üzerinde ardarda gelen verilere ait elektrik sinyalleri aynı anda yol alırlar. Elektrik sinyalleri, kullanılan iletken malzemenin yayılım gecikmesi sayesinde birbirlerini bozmazlar. Bu sinyaller belli aralıklar ile kanala ulaşırlar ve kanal tarafından cevap olarak gerekli sinyaller aktif hale getirilir. Bu çalışma prensibi, veri haberleşmesinde kullanılan kayan pencere (sliding window) protokolüne benzetilebilir. Her veri için kanaldan cevap beklemek yerine aradaki bağlantının güvenirliği göz önüne alınarak birden fazla istek ardarda yapılır. Verinin kanaldan kontrol birimine aktarıldığı yazma türü komutlarda, kontrol birimi belli bir periyodda kanala yeni veri istediğini bildirir. Kanal ise bu isteklere cevap olarak BusOut yoluna yeni veriyi çıkarır ve gerekli sinyali aktif hale getirir.

Veri akışı yönteminde oluşabilecek hata durumlarını belirleyebilmek için bir zamanlama yöntemi oluşturulmuştur. Bu yönteme göre kontrol birimi en son aktif hale getirdiği Service In veya Service Out sinyaline ait cevabı 8 ms içerisinde alamazsa kanal ile bağlantıda bir problem oluştuğuna ve yürütülmekte olan işlemin durdurulmasına karar verir.

4. PARALEL KANAL İLE PCI YEREL YOLUNUN BİRLEŞTİRİLMESİ VE PCI UYUMLU ÇEVRE BİRİMLERİNİN ANA BİLGİSAYAR SİSTEMLERİNE KULLANDIRILMASI

Bu tez çalışmasında, ana bilgisayar sistemlerinin, paralel kanal giriş_çıkış bağlantısı ile kişisel bilgisayar sistemlerindeki PCI yerel yolunun birleştirilmesi ve kişisel bilgisayar çevre birimlerinin ana bilgisayar sistemlerine kullandırılması için bir sistem gerçekleştirilmiştir. Bu sistemin yapısı, veri haberleşmesinde farklı topolojiler ve protokoller arasında geçiş sağlanmasında uygulanan yöntemlere benzer şekilde modellenmiştir.

Yapılan modelleme çalışması sonunda sistemin sahip olması gereken özellikler aşağıdaki gibi belirlenmiştir.

1. Sistem, kolay tasarlanabilir, gerçekleştirilebilir, genişletilebilir ve test edilebilir olması amacıyla modüler katmanlarla gerçeklenmelidir.

2. Katmanlar arasında, donanım ve yazılımlardan yararlanılarak, özellikleri bu tez çalışması kapsamında belirlenen standart yapılara uygun veri alışverişi sağlanmalıdır.

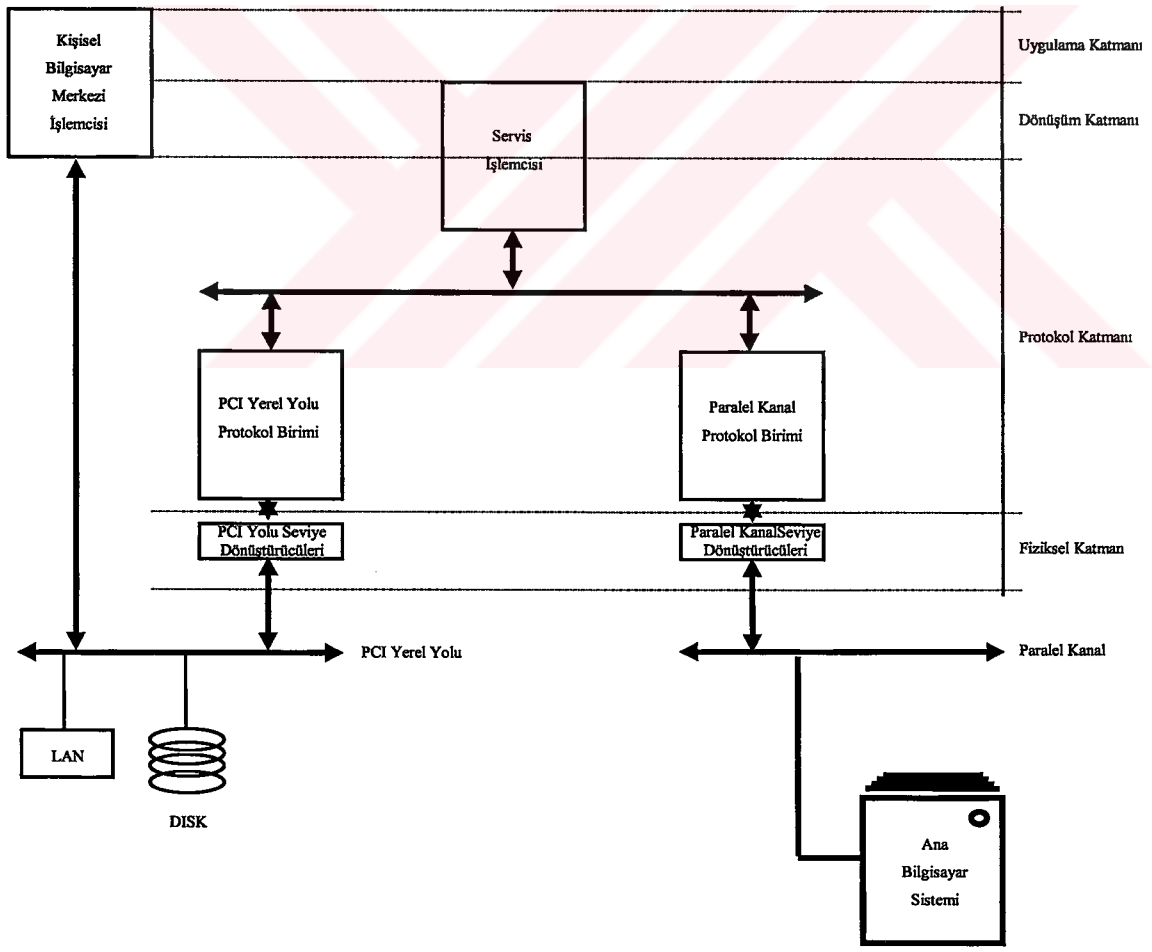
3. Donanım ile gerçekleştirilen kısımlarda, problem mümkün olduğunca küçük parçalara bölme yöntemi ile algoritmik olarak ifade edilmeli, yazılım eşdeğeri oluşturulup testleri tamamlandıktan sonra gerekli araçlar kullanılarak programlanabilir donanıma aktarılmalıdır.

4. Yazılım ile gerçekleştirilen kısımlarda, paralel ve dağıtılmış çalışmayı sağlayabilmek için bağlı bulunan kişisel bilgisayar sisteminin merkezi işlemcisi dışında bir servis işlemcisi bulunmalıdır. Merkezi işlemci ve servis işlemcisi arasında, yürütülecek işlemler paylaştırılmalı, aynı anda birden fazla görevin yerine getirilmesi sağlanmalıdır.

Modelleme sonucunda belirlenen özelliklere göre oluşturulan sistemin,

1. Fiziksel katman
2. Protokol katmanı
3. Dönüşüm katmanı
4. Uygulama katmanı

olarak isimlendirilen 4 katmandan oluşmasına karar verilmiştir. Bu katman yapısına uygun olarak gerçekleştirilen sistemin blok diyagramı ve katmanların sınırları ile aralarındaki bağlantı Şekil 4.1'de görülmektedir.



Şekil 4.1 Paralel kanal ile PCI yerel yolu bağlantısını sağlayan sistemin katman yapısı

Fiziksel katman, paralel kanala ve PCI yerel yoluna elektriksel bağlantıyı sağlayan seviye dönüştürücülerinden oluşur. PCI yerel yolu protokol birimi ve paralel kanal protokol birimi ile servis işlemcisi, protokol katmanını oluştururlar. Servis işlemcisi ve merkezi işlemcide çalışan yazılımlar dönüşüm katmanı olarak görev yaparlar. Uygulama katmanı ise merkez işlemcide gerçeklenir.

Bu katmanların yapıları ve içerdikleri fonksiyonlar aşağıdaki alt başlıklarda ayrıntılı olarak incelenmiştir.

4.1 Fiziksel Katman

Fiziksel katman, paralel kanala ve PCI yerel yoluna elektriksel bağlantıyı sağlayan seviye dönüştürücülerinden oluşur.

4.1.1 Fiziksel katmanın tanımı

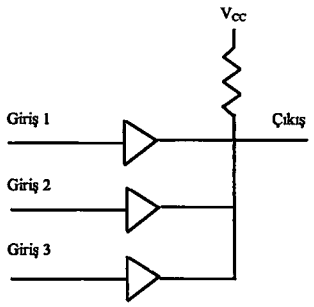
Fiziksel katman, gerçekleştirilen sistemin, paralel kanal ve PCI yerel yolu ile elektriksel bağlantısını sağlar. Gerçekleştirilen sistem kendi içerisinde TTL (Transistor-Transistor-Logic) lojik seviyesindeki sinyaller ile çalıştığı için, paralel kanal ve PCI yerel yolundan gelen sinyallerin TTL seviyesine dönüştürülmesi gereklidir. Aynı şekilde, sistemden çıkacak sinyallerin de gideceği yolun elektriksel özelliğine uygun hale getirilmesi zorunludur.

4.1.2 Fiziksel katmanın paralel kanal ile bağlantısı

Paralel kanalda tanımlı lojik seviyeler $L(v)$ ile gösterilecek olursa, bu lojik seviyelere karşılık gelen gerilimler aşağıdaki gibi tanımlanmıştır:

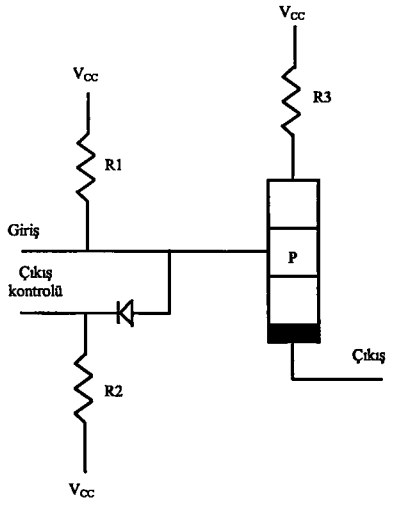
$$L(v)=\begin{cases}1 \rightarrow & 2.25 \le v \le 6 \\ 0 \rightarrow & -0.15 \le v \le 0.15\end{cases} \qquad (4.1)$$

Kanal yapısında sinyallerin aktarımı telli-veya (wired-or, dot-or) prensibine göre gerçekleşir (Şekil 4.2). Telli-veya yapısında çıkış yönünde tanımlı olan sinyaller, sürücü devrelerinden geçtikten sonra fiziksel olarak birbirlerine bağlanırlar. Bu sürücülerden birinin çıkışının aktif duruma gelmesi, hattı aktif duruma getirir.



Şekil 4.2 Telli-veya (wired-or, dot-or) bağlantısı

Hattaki alıcılar, giriş sinyali aktif hale geldiğinde, bu sinyali hangi sürücünün aktif hale getirdiğine bakmaksızın, çıkışlarına hattın aktif olduğu bilgisini aktarırlar. Telli-veya fonksiyonunu gerçekleştirmek için kullanılan sürücülerin prensip şeması Şekil 4.3'de görülmektedir. Bu sürücülerin çıkışı uygun alıcılar ile TTL seviyesine dönüştürülmüştür. Sistemden paralel kanala giden sinyaller de aynı şekilde uygun sürücüler ile kanalda kullanılan lojik seviyelere dönüştürülmüştür.



Şekil 4.3 Paralel kanalda kullanılan telli veya yapısını destekleyen sürücüler

4.1.3 Fiziksel katmanın PCI yerel yolu ile bağlantısı

PCI yerel yolu, hem 5V hem de 3.3V gerilim seviyelerine uygun lojik sinyaller ile çalışır. Besleme gerilimi V_{cc} ile ifade edildiğinde,

$$4.75 \le V_{cc} \le 5.25$$

şartının sağlandığı durumlarda giriş gerilimi v'ye karşılık gelen lojik seviyeleri belirten $L(v)$ bağıntısı aşağıdaki gibidir:

$$L(v) = \begin{cases} 1 \rightarrow & 2.0 \le v \le V_{cc} + 0.5 \\ 0 \rightarrow & -0.5 \le v \le 0.8 \end{cases} \qquad (4.2)$$

Besleme geriliminin,

$$3.0 \le V_{cc} \le 3.6$$

şartını sağlaması durumunda, giriş gerilimi v ile çıkışta oluşan lojik seviyeler arasındaki $L(v)$ bağıntısı şu şekilde ifade edilir:

$$L(v) = \begin{cases} 1 \rightarrow & \frac{1}{2}V_{cc} \le v \le V_{cc} + 0.5 \\ 0 \rightarrow & -0.5 \le v \le \frac{1}{3}V_{cc} \end{cases} \qquad (4.3)$$

Gerçekleştirilen sistem, 5V gerilim seviyesine uygun PCI yerel yolları ile çalışmak üzere tasarlanmıştır. 5V ile çalışmada geçerli olan gerilim seviyeleri TTL gerilim seviyelerine uyum sağladığı için arada özel gerilim seviyesi dönüştürcülerine ihtiyaç duyulmamıştır. 3.3V gerilim seviyesine uygun çalışan PCI yerel yolları için, gerçekleştirilen sistemin

yapısında değişikliğe gidilmeden araya gerilim seviyesi dönüştürücülerinin konulması yeterlidir.

4.2 Protokol Katmanı

PCI yerel yolu protokol birimi ve paralel kanal protokol birimi ile servis işlemcisi, protokol katmanını oluştururlar.

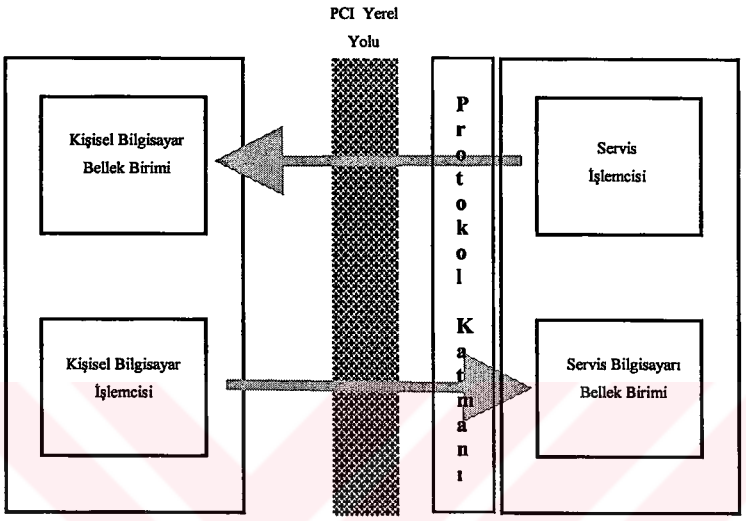
4.2.1 Protokol katmanının tanımı

Protokol katmanı, fiziksel katmandan gelen sinyalleri değerlendiren ve bu sinyalleri anlamlı hale getiren katmandır. Paralel kanalda ve PCI yerel yolunda işlem yapabilmek için belirli kurallara ve zamanlamalara uyulması gereklidir. Protokol katmanı her iki yola da, kurallarına ve zamanlamalarına uygun erişilmesini ve bu yollarda yer alan birimlerle haberleşmeyi sağlar. Paralel kanala veya PCI yerel yoluna erişim hakkının elde edilmesi, işlem yapılacak birimlere ait adreslerin oluşturulması ve çözülmesi, okuma ve yazma yönünde veri aktarımının başlatılması, sürdürülmesi, sonlandırılması ve işlemler sonucunda oluşan durum bilgisinin üretilmesi ile hata durumlarında tanımlı işlemlerin yerine getirilmesi protokol katmanının görevleridir. Protokol katmanının görevleri, yazılım ve donanım ile gerçekleştirilmiştir ve bu katman, diğer katmanlar tarafından saklayıcılar üzerinden erişilecek şekilde tasarlanmıştır. Protokol katmanının yazılım ile gerçeklenen görevleri, servis işlemcisi tarafından yürütülür.

4.2.2 Protokol katmanının PCI yerel yoluna erişimi

Gerçekleştirilen protokol katmanı, PCI yerel yoluna ana birim olarak erişebildiği gibi, aynı yola bağlı diğer birimlerin kendisini hedef birim olarak adreslemelerine ve erişmelerine de izin verir. Bu özellik sayesinde, gerçekleştirilen sisteme ait saklayıcı ve bellek birimleri ile PCI yerel yolunda bulunan diğer saklayıcı ve bellek birimleri arasında saydam paylaşım yapılabilmektedir. Servis işlemcisinin ve kişisel bilgisayardaki merkezi işlemcinin, kendilerinde olmayan saklayıcı veya bellek birimlerine, protokol katmanında

gerçekleştirilen tasarım sayesinde adres dönüşümü yöntemiyle, kendi adresleme alanları üzerinden erişmeleri sağlanmıştır (Şekil 4.4).



Şekil 4.4 Servis işlemcisi ile merkezi işlemci arasındaki bağlantı

4.2.2.1 Protokol katmanının PCI yerel yolunda ana birim olması

Protokol katmanı, PCI yerel yolunda yer alan diğer kaynaklara ana birim olarak erişmek istediğinde PCI yerel yolu protokolüne uygun olarak seçme işlemine katılacak ve erişim hakkı kendisine verildiğinde başlattığı erişimi tamamlayacak şekilde gerçekleştirilmiştir. Bu sayede üst seviyedeki katmanlar, PCI yerel yoluna erişmek istediklerinde bu isteklerini protokol katmanına aktarıp erişimin detayları ile ilgilenmek durumunda kalmazlar.

4.2.2.2 Protokol katmanının PCI yerel yolunda hedef birim olması

Protokol katmanı, PCI yerel yolu üzerinde hedef birim olarak davranabilmek için kendisine atanmış bir adres bölgesine ihtiyaç duymaktadır. Bu adres bölgesi, PCI yerel yolunun

otomatik yapılandırma özellikleri de göz önüne alınarak yazılım tarafından değiştirilebilir şekilde dinamik olarak belirlenebilmektedir. Gerçekleştirilen sistemde, protokol katmanı PCI yerel yolu üzerinde hedef birim olarak hem bellek adres bölgesinde hem de giriş_çıkış adres bölgesinde çalışabilecek şekilde tasarlanmıştır.

4.2.3 Protokol katmanının dönüşüm katmanına sunduğu PCI yerel yolu servisleri

Protokol katmanında, dönüşüm katmanını oluşturan servis işlemcisi ile merkezi işlemci arasında, kaynakların paylaşılması durumunda oluşabilecek çakışmaları engellemek ve senkronizasyonu sağlamak için semafor görevini yerine getirecek özel saklayıcılara yer verilmiştir.

Protokol katmanında, birbirlerinden bağımsız olarak çalışacak şekilde tasarlanan servis işlemcisi ile merkezi işlemci arasında veri aktarımı için 2 yöntem tanımlanmıştır:

Aktarılacak verinin byte cinsinden uzunluğu l ile ifade edilecek olursa,

1. $l \leq 32$ için protokol katmanında yer alan ve iki yönde veri aktarımı imkanı sağlayan özel saklayıcılar kullanılmalıdır.

2. $l > 32$ için doğrudan bellek paylaşımı yöntemi uygulanmalı ve sistem performansını arttıracak şekilde mümkün olduğunca anlık aktarımlarla yapılmalıdır.

4.2.4 Protokol katmanının paralel kanala erişimi

Protokol katmanı paralel kanala erişim için gerekli olan fonksiyonları da içermektedir. Bu fonksiyonların gerçekleştirilmesinde hem donanım hem de yazılım yöntemlerinden yararlanılmıştır. Özellikle zamanlama açısından hassas değerlere bağlı kalınmasını ve belli sürelerde cevap oluşturulmasını gerektiren fonksiyonlar donanım ile, bu fonksiyonların başlatılıp durdurulması ise yazılım ile gerçeklenmiştir.

4.2.5 Protokol katmanında birden fazla paralel kanal giriş_çıkış biriminin desteklenmesi

Yapılan tasarımda protokol katmanının, aynı anda farklı tipte kontrol birimlerini desteklemesi ve kontrol birimi tiplerinin yazılım ile belirlenebilmesi sağlanmıştır. Protokol katmanı, kontrol birimleri ile bu birimlerin yönettiği giriş_çıkış birimleri arasında ayırım yapmayacak şekilde tasarlanmıştır. Protokol katmanının paralel kanalda cevap vereceği ve istek oluşturacağı adresler yazılım ile belirlenecek şekilde gerçekleştirilmiştir.

Mevcut sistemler incelendiğinde, bunların tasarım aşamasında belirlenmiş ve sonradan değiştirilemeyen kontrol birimlerini destekledikleri ve aynı anda farklı kontrol birimlerine izin vermedikleri görülmüştür. Gerçekleştirilen sistem, tanımlanabilecek kontrol birimi tipi açısından da bir sınır getirmediği için mevcut sistemlere üstünlük sağlamaktadır.

4.2.6 Protokol katmanı ile paralel kanal arasında veri aktarımı

Paralel kanalda, aktarım hızı sırasıyla

1. karşılıklı kilitlemeli (interlocked) veri aktarımı
2. yüksek hızlı (high speed) veri aktarımı
3. veri akışı (data streaming) ile veri aktarımı

yöntemi olmak üzere 3 farklı aktarım yöntemi tanımlanmıştır. Protokol katmanı, tüm bu veri aktarım yöntemlerini destekleyecek şekilde tasarlanmıştır. Ana bilgisayar sistemine kullandırılacak giriş_çıkış birimlerinin veri aktarım hızlarına bağlı olarak sistemin hangi aktarım yöntemini kullanacağı yazılım aracılığı ile ayarlanabilmektedir. Sistem çalışması durdurulmadan aktarım yöntemleri arasında geçiş yapma imkanı da tanınmıştır. Farklı tipte kontrol birimlerinin aynı anda tanımlanmış olması durumunda her kontrol birimi için ayrı bir veri aktarım yöntemi seçilebilmesi mümkün olmaktadır. Gerçekleştirilen sistem, bu

özellikleri sayesinde, tek bir kontrol biriminin sabit bir veri aktarım yöntemi ile uygulandığı mevcut sistemlere göre daha üstündür.

4.2.7 Protokol katmanı ile paralel kanal arasında veri akışı yönteminin uygulanması

Protokol katmanının önemli bir özelliği de veri akışı yöntemi ile veri aktarımını desteklemesi ve bu yöntemi yazılım destekli donanım ile gerçekleştirilmiş olmasıdır.

Veri akışı yöntemi, elektrik sinyallerinin iletilmeleri sırasında karşılaşılan yayılım gecikmesinden yararlanılarak, aynı hat üzerinde belirli aralıklarla farklı elektriksel sinyallerin yollanması prensibine dayanmaktadır. Bu durumda iletken hat üzerinde aynı anda farklı elektriksel sinyal seviyeleri bulunmaktadır. Sinyaller vericiden çıktıktan belli bir süre sonra, çıkış gecikmeleri kadar farklar ile alıcı tarafa ulaşmaktadırlar. Alıcı taraf, doğru olarak aldığı her sinyal için vericiye bir cevap sinyali yollamaktadır. Verici taraf, yeni bir sinyal yollayacağı zaman, bir önce yolladığı sinyal için cevap sinyali beklemediği için, ardarda belli aralıklarla sinyal çıkarabilmekte ve hat üzerinde ilerleyen sinyaller ile bir iş hattı oluşturmaktadır.

Veri haberleşmesinde, yayılım gecikmesinin yüksek olduğu hatlarda verimi arttırmak amacıyla uygulanan kayan pencere protokolüne benzeyen bu yöntem sayesinde kanal ile kontrol birimi arasında daha hızlı veri aktarımı yapılabilmektedir. Kayan pencere protokolünde her pencere için bir onay bilgisi kullanılırken, kanal protokolünde doğru alınan her sinyal için bir onay sinyali yollanmaktadır. Verici taraf, gönderdiği sinyal sayısı ile aldığı onay sinyali sayısını karşılaştırarak bilgilerin doğru olarak aktarılıp aktarılmadığını kontrol etmektedir. İş hattı yapısı sebebiyle onay sinyalleri, gerçek sinyallerin yollanmasının tamamlanmasından bir süre sonrasına kadar devam etmektedir. Protokol katmanında veri akışı yöntemine göre veri aktarımını sağlamak amacıyla programlanabilir lojik kullanılması öngörülmüştür.

4.3 Dönüşüm Katmanı

Servis işlemcisi ve merkezi işlemcide çalışan yazılımlar dönüşüm katmanı olarak görev yaparlar.

4.3.1 Dönüşüm katmanının tanımı

Dönüşüm katmanı, kişisel bilgisayar sistemlerindeki giriş_çıkış birimleri ile ana bilgisayar sistemlerindeki giriş_çıkış birimleri arasındaki yapısal ve fonksiyonel farklılıkları ortadan kaldırmak amacıyla gerçeklenmiştir. Dönüşüm katmanı, hem servis işlemcisinde hem de merkezi işlemcide gerçeklenmiş iki bölümden oluşmaktadır.

4.3.2 Ana bilgisayar sistemlerinde giriş_çıkış yapısı

Ana bilgisayar sistemlerinde dağıtılmış bir giriş_çıkış yapısı vardır. Giriş_çıkış birimleri, ana bilgisayar sistemine arada yer alan bağımsız işlem yeteneğine sahip kontrol birimleri üzerinden bağlıdırlar. Kontrol birimleri, ana bilgisayar sistemi ile giriş_çıkış birimleri arasında bir köprü görevi görürler ve giriş_çıkış birimlerinin tasarımlarından, iç yapılarından ve çalışma şekillerinden kaynaklanabilecek farklılıkları paralel kanaldan gizlerler. Bu sayede ana bilgisayar sistemi bütün giriş_çıkış birimlerine aynı programlama modelini kullanarak erişebilir. Bu programlama modelinde, kontrol birimleri ve giriş_çıkış birimleri için değeri 0 ile 255 arasında değişen giriş_çıkış adresleri ve okuma, yazma ve kontrol olmak üzere 3 temel işlem tanımlanmıştır.

Kanal tarafından yürütülmesi istenen giriş_çıkış işlemleri, çevre birimleri ile kanal arasında yer alan kontrol birimleri sayesinde, dağıtılmış ve paralel olarak gerçekleştirilirler. Kontrol birimleri, kanal tarafından giriş_çıkış işleminin başlatılmasından bu işlemin sonuçlanmasına kadar geçen süre içerisinde, giriş_çıkış biriminde işlemin yürütülmesinden, kanal ile okuma veya yazma yönünde veri aktarımından ve işlemin sonlandırılmasından sorumludurlar.

4.3.3 Kişisel bilgisayar sistemlerinde giriş_çıkış yapısı

Kişisel bilgisayar sistemlerinde özel uygulamaların dışında, ana bilgisayar sistemlerinin aksine giriş_çıkış birimlerini kontrol etmekle görevli bağımsız işlem gücüne sahip kontrol birimleri bulunmaz. Merkezi işlemci, çevre birimlerinde gerçekleştirilecek giriş_çıkış işlemlerini başlatmaktan, yürütmekten ve sonlandırmaktan sorumludur. Kişisel bilgisayar sistemlerinde, kendi belleği ve işlemcisi bulunan ve merkezi işlemciden bağımsız çalışma yeteneği olan kontrol birimlerinin yeterince yaygınlaşmamış olması, bu sistemlerde giriş_çıkış birimlerine erişmede kullanılacak ortak bir yazılım modeli oluşturulamamasına sebep olmuştur. Mevcut kişisel bilgisayar giriş_çıkış yapıları incelendiğinde, bu tarz akıllı kontrol birimlerine ancak yüksek hızlı veri aktarım özelliğine sahip disk veya teyp birimlerinin kişisel bilgisayar sistemine bağlantılarında rastlanmaktadır. Çalışma prensibi olarak, paralel kanal yapısına pek çok yönden benzeyen SCSI (Small Computer Systems Interface) arabirimi bu tarz yapılardan biridir.

Paralel kanal yapısına bağlanan çevre birimlerinin aksine, kişisel bilgisayar sistemlerindeki çevre birimlerinin, giriş_çıkış işlemleri yürütmek için programlanmasında izlenmesi gereken yöntem, bu birimlerin iç yapılarına bağlıdır. Her çevre birimi, özelliğine bağlı olarak, kendisi ile merkezi işlemci arasında haberleşmeyi sağlayan değişik sayıda iç saklayıcıya sahiptir. Giriş_çıkış işlemlerinin başlatılması ve yürütülmesi, merkezi işlemcinin çevre birimlerine bu saklayıcılar üzerinden erişmesi ve çevre birimine özgü komutları aktarması ile gerçekleşir. Çevre birimleri de giriş_çıkış işlemlerinde oluşan değişik durumları ve merkezi işlemcinin müdahalesine ihtiyaç duyduklarını bu saklayıcılar aracılığı ile belirtirler.

4.3.4 Dönüşüm katmanının soyutlama işlevi

Ana bilgisayar ve kişisel bilgisayar sistemlerinin çevre birimleri arasında yukarıda belirtilen farklara ek olarak aynı görevi yerine getirmek üzere tasarlanmış çevre birimleri arasında bile iç yapıları, sahip oldukları saklayıcılar, kabul ettikleri komutlar ve

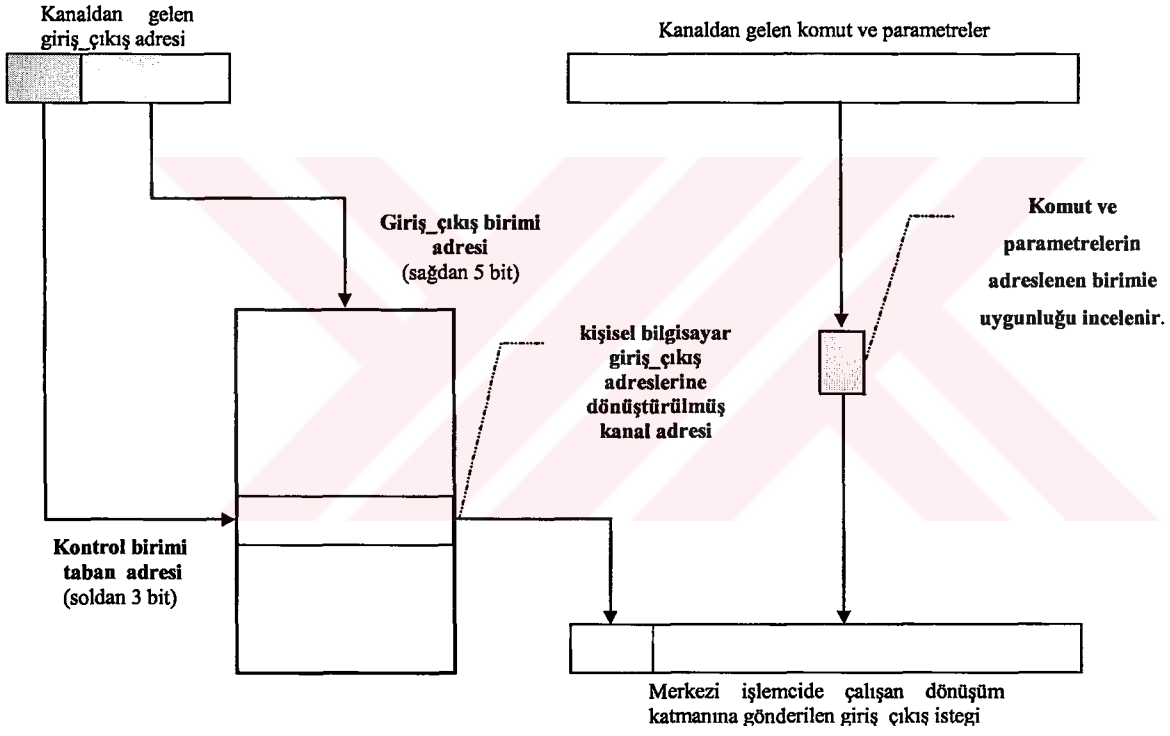
döndürdükleri cevaplar açısından farklılıklar olduğu göz önüne alınarak kişisel bilgisayarlardaki çevre birimlerine erişimde ortaya çıkacak problemleri gidermek için bir soyutlama katmanına (abstraction layer) ihtiyaç olduğu belirlenmiştir.

Dönüşüm katmanı sayesinde kişisel bilgisayar sisteminde yer alan çevre birimlerine yapısal farklılıklarından bağımsız, genel bir programlama modeli ile erişilmesi ve ana bilgisayar sisteminden gelen giriş_çıkış isteklerinin bu çevre birimlerine yönlendirilmesi sağlanmıştır. Dönüşüm katmanının içerdiği fonksiyonlar, yazılım ile hem servis işlemcisinde hem de kişisel bilgisayardaki merkezi işlemcide gerçeklenmiştir. Soyutlama işlemi, dönüşüm katmanı ile uygulama katmanı arasındaki iki yönlü geçiş ile gerçekleştirilmiştir.

4.3.5 Dönüşüm katmanının servis işlemcisinde uygulanan kısmı

Dönüşüm katmanının, servis işlemcisinde gerçekleştirilen kısmı, paralel kanal ile ilgili işlemleri yerine getirmektedir. Paralel kanaldan gelen giriş_çıkış birimi seçme istekleri, protokol katmanı tarafından dönüşüm katmanına aktarılır. Dönüşüm katmanı, bu isteklerdeki adresi, kendisine yazılım ile tanıtılmış ve çalışma sırasında da değiştirilebilen adresler ile karşılaştırır. Gelen adresin dönüşüm katmanına tanıtılmış adresler içinde bulunması durumunda, protokol katmanının bu seçme isteğine olumlu cevap vermesi sağlanır ve seçme isteğinin tamamlanabilmesi için, kanala, protokol katmanı üzerinden ilk durum bilgisi yollanır. Seçme isteği ile gelen giriş_çıkış komutu incelenerek, adreslenen giriş_çıkış kontrol birimine uygunluğu ve geçerliliği belirlenir. Ek parametrelere ihtiyaç duyulması durumunda kanal ile bağlantı devam ettirilerek bu parametrelerin kanaldan alınması sağlanır. Servis işlemcisi, paralel kanaldan gelen giriş_çıkış isteğinin, kişisel bilgisayar sisteminde hangi giriş_çıkış birimi kullanılarak gerçekleştirileceğini, içeriği sistemin çalışması sırasında yazılım ile dinamik olarak değiştirilebilen dönüşüm tablosundan belirler. Komut ve komuta ait parametrelerle kişisel bilgisayar sisteminde hangi giriş_çıkış biriminin kullanılacağı, dönüşüm katmanının merkez işlemci üzerinde çalışan kısmına protokol katmanı üzerinden aktarılır. Parelel kanaldan gelen bir isteğin dönüştürülmesine ait akış şeması Şekil 4.5'de görülmektedir.

Yapılan incelemeler sonunda paralel kanal mimarisinde tanımlı giriş_çıkış komutlarının parametreleri ile birlikte 25 byte'ı aşmadıkları belirlenmiş ve merkezi işlemciye komut ve parametre aktarımının, protokol katmanındaki 32 byte'lık özel çift yönlü arabellek ile gerçekleştirilmesine karar verilmiştir. Giriş_çıkış işlemlerine ait verilerin aktarımı ise protokol katmanı tarafından sağlanan bellek paylaşımı yöntemi ile gerçekleştirilmektedir. Dönüşüm katmanı yazma türü komutlarda kanaldan gelen veriyi uygulama katmanının da erişebileceği ana bilgisayar sistemi adres alanı içerisindeki ara belleklere aktarır. Okuma yönünde gerçekleşen komutlarda ise, uygulama katmanı giriş_çıkış biriminden gelen bilgiyi, kişisel bilgisayar adres alanı içerisinde dönüşüm katmanı ile paylaşılabilen ara bellek bölgelerine yazar.



Şekil 4.5 Paralel kanaldan gelen giriş_çıkış adresinin kişisel bilgisayar giriş_çıkış adresine dönüştürülmesi

4.3.6 Dönüşüm katmanının merkezi işlemcide uygulanan kısmı

Dönüşüm katmanının merkezi işlemci üzerinde gerçekleştirilen kısmı, servis işlemcisinden kendisine gelen komutları ve parametreleri değerlendirerek bunları, kişisel bilgisayar giriş_çıkış birimlerinde çalışmaya uygun hale dönüştürmek ve bu birimlerin

kapasitelerinden, fiziksel özelliklerinden ve çalışma prensiplerinden kaynaklanabilecek farklılıkları ortadan kaldırmak için oluşturulmuştur. Komut ve parametrelerin kişisel bilgisayar sistemi giriş_çıkış birimlerine uygun hale getirilmesinden sonra dönüşüm katmanı, bu giriş çıkış isteğini çalıştırılmak üzere uygulama katmanına yollar.

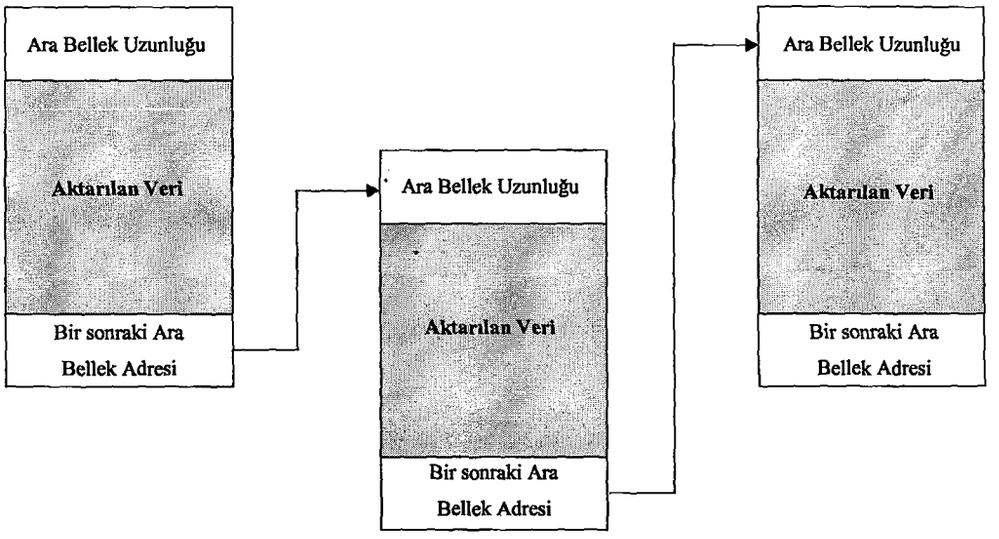
Uygulama katmanında tamamlanan giriş_çıkış işlemlerinin sonucu dönüşüm katmanına aktarılır. İşlemlerin tamamlanması 2 şekilde olabilir:

1. Veri aktarımı gerektirmeyen işlemler için, dönüşüm katmanına sadece sonlanma bilgisi döndürülür ve dönüşüm katmanı, bu sonlanma bilgisini ana bilgisayar sistemine uygun hale getirir.

2. Veri aktarımı gerektiren giriş_çıkış işlemleri, bir veya birden fazla aktarım işleminden oluşabilirler. Veri aktarımının yönüne bağlı olarak, ya dönüşüm katmanı kanaldan aldığı veriyi uygulama katmanına ya da uygulama katmanı giriş_çıkış biriminden aldığı veriyi dönüşüm katmanına paylaşılan ara bellekler üzerinden aktarır.

Birden fazla aktarımın gerektiği giriş_çıkış işlemlerinde sistemin performansını arttırmak ve katmanların birbirlerini beklemelerini en aza indirgemek için, veri aktarımında kullanılan ara bellekler birbirlerine linkli liste yapısı ile bağlanmışlardır. Bu arabellek yapısı Şekil 4.6'da görülmektedir.

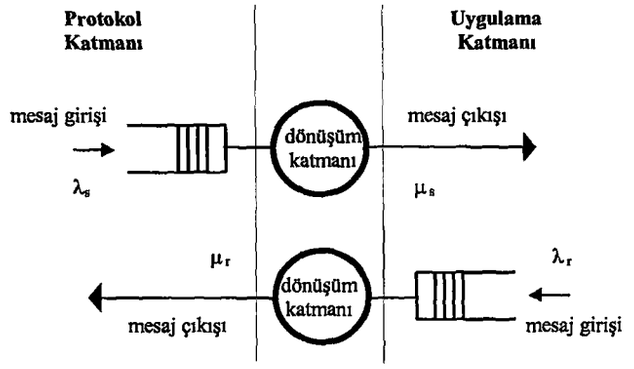
Dönüşüm katmanı, kişisel bilgisayar sistemine bağlı giriş_çıkış birimlerinde oluşan durum değişikliklerini de paralel kanala aktaracak şekilde tasarlanmıştır. Giriş_çıkış birimleri, yeni veri gelmesi, yazma işleminin tamamlanması veya hazır duruma geçilmesi gibi durumlarda, kendilerinde olan değişiklikleri, uygulama katmanına kesme istekleri ile belirtirler. Uygulama katmanına gelen bu istekler, dönüşüm katmanına iletilirler ve dönüşüm katmanı, isteğin geldiği kişisel bilgisayar giriş_çıkış birimine, ana bilgisayar sisteminde hangi adresin denk düştüğünü belirler ve paralel kanalda bir seçme işlemi gerçekleştirilmesi için gerekli isteği protokol katmanı üzerinden paralel kanala yollar.



Şekil 4.6 Veri aktarımında, dönüşüm katmanı ile uygulama katmanı arasında kullanılan arabellek yapısı

4.3.7 Dönüşüm katmanının istatistiki modeli

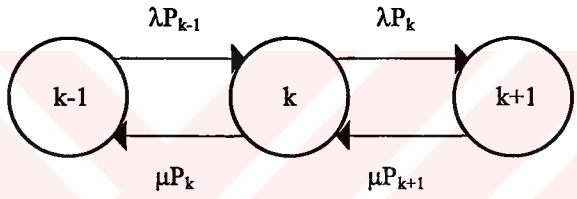
Dönüşüm katmanı, Şekil 4.7'de görüldüğü gibi, girişine gelen istekleri işledikten sonra çıkışına aktaran bir yapı olarak gerçeklenmiştir. Paralel kanaldan kişisel bilgisayar çevre birimlerine dönüşümde protokol katmanı girişi, uygulama katmanı ise çıkışı oluştururlar. Kişisel bilgisayar çevre birimlerinden paralel kanala dönüşümde ise uygulama katmanı girişi, protokol katmanı ise çıkışı meydana getirirler.



Şekil 4.7 Dönüşüm katmanındaki kuyruk yapısı

Katmanlar arasında aktarılan bilgileri, uzunluğu belirli mesajlar olarak değerlendirecek olursak, dönüşüm katmanına, birim zamanda gelen ortalama mesaj sayısı ile dönüşüm katmanından birim zamanda çıkan ortalama mesaj sayısı birbirlerine eşittir. Bu özelliği ile dönüşüm katmanı, durağan bir sistem oluşturmaktadır ve "birth-death" modeline uygun davranmaktadır.

Dönüşüm katmanındaki giriş debisi λ, çıkış debisi μ ve kuyrukta k sayıda mesaj olması olasılığı P_k ile gösterilecek olursa kuyruğa mesajların girmesi ve çıkması arasındaki ilişki Şekil 4.8'deki gibidir:



Şekil 4.8 Dönüşüm katmanında kuyruk uzunluğunu gösteren durumlar arasındaki geçişler

Şekil 4.8'de, k ve $k+1$ durumları arasında dengenin sağlanması

$$\lambda \ P_k = \mu \ P_{k+1} \qquad (4.4)$$

şartına bağlıdır ve dönüşüm katmanındaki trafik yoğunluğu ρ

$$\rho = \frac{\lambda}{\mu} \qquad (4.5)$$

bağıntısı ile hesaplanır. İlk k durum için denge bağıntıları yazılacak olursa

$$\begin{aligned} \lambda\ P_0 &= \mu\ P_1 \\ \lambda\ P_1 &= \mu\ P_2 \\ \lambda\ P_2 &= \mu\ P_3 \\ &\vdots \quad\ \vdots \\ \lambda\ P_{k-1} &= \mu\ P_k \end{aligned} \qquad (4.6)$$

denklem takımından kuyrukta k mesaj olma olasılığını gösteren P_k için

$$P_k = \rho^k P_0 \qquad (4.7)$$

bağıntısı elde edilir. Bu bağıntıdaki bilinmeyen P_0 değerini hesaplamak amacıyla

$$\sum_{k=0}^{\infty} \rho^k P_0 = 1 \qquad (4.8)$$

eşitliğinden yararlanılacak olursa, P_0 için

$$P_0 = 1 - \rho$$

değeri elde edilir. Bu durumda P_k

$$P_k = (1-\rho)\rho^k \qquad (4.9)$$

bağıntısı ile ifade edilir.

Dönüşüm katmanında herhangi bir t anında kuyrukta bekleyen ortalama mesaj miktarı $\overline{m}$ değerini bulmak amacıyla

$$\overline{m} = \sum_{k=0}^{\infty} kP_k = (1-\rho)\sum_{k=0}^{\infty} k\rho^k$$

eşitliği yazılacak olursa

$$\overline{m} = \frac{\rho}{1-\rho} \qquad (4.10)$$

bağıntısı elde edilir.

Dönüşüm katmanının bir mesaja servis verme süresi τ_s ile belirtilirse, kuyrukta ortalama bekleme zamanı τ_w değeri $\overline{m}$ ve τ_s cinsinden aşağıdaki bağıntı ile ifade edilir:

$$\tau_w = \overline{m}\,\tau_s = \frac{\rho}{1-\rho}\tau_s = \frac{\rho}{1-\rho}\frac{1}{\mu} \qquad (4.11)$$

Dönüşüm katmanı tarafından bir mesajın aktarılması için gerekli olan τ_r süresi ise

$$\tau_r = \tau_w + \tau_s \qquad (4.12)$$

eşitliği ile hesaplanabilir. Bu eşitliğin açılması ile önce

$$\tau_r = \frac{\tau_s}{1-\rho} \qquad (4.13)$$

bağıntısına erişilir ve buradan bağıntının sağ tarafının $\frac{\lambda}{\lambda}$ çarpılması ile τ_r değeri, $\overline{m}$ ve λ değerleri cinsinden

$$\tau_r = \frac{\overline{m}}{\lambda} \qquad (4.14)$$

bağıntısı ile elde edilir.

4.4 Uygulama Katmanı

Merkezi işlemcide çalışan yazılımlar uygulama katmanı olarak görev yaparlar.

4.4.1 Uygulama katmanının tanımı

Uygulama katmanı, dönüşüm katmanından gelen giriş_çıkış isteklerini, kişisel bilgisayar sisteminde ilgili çevre birimlerinde başlatmak, yürütmek, sonlandırmak ve dönüşüm katmanının bu çevre birimlerinin iç yapılarından bağımsız çalışabilmesini sağlamak amacıyla gerçekleştirilmiştir.

4.4.2 Uygulama katmanının özellikleri

Uygulama katmanı, kişisel bilgisayar sistemine ait çevre birimlerinde, paralel kanaldan gelen giriş_çıkış işlemlerinin gerçekleştirildiği katmandır. Uygulama katmanı, çevre birimlerine, bu birimlere ait cihaz sürücüler üzerinden erişecek şekilde tasarlanmıştır.

Aynı anda farklı cihaz sürücüleri üzerinden çevre birimlerinde giriş_çıkış işlemlerinin yürütülebilmesi için, uygulama katmanı, kişisel bilgisayar sisteminde çalışan işletim sisteminin sağladığı çoklu programlama (multiprogramming) modelinden yararlandırılmıştır. Uygulama katmanında, işlem parçacığı (thread) yapısının desteklenmesi sayesinde, gelen her giriş_çıkış isteği için bir işlem parçacığı oluşturulur. Cihaz sürücüleri ile haberleşerek giriş_çıkış işlemlerinin gerçekleştirilmesi oluşturulan bu işlem parçacıkları tarafından yürütülür. Dönüşüm katmanından gelen istek yapısı, uygulama katmanında çok sayıda işleme gerek kalmadan isteğin hangi cihaz sürürücüye yönledirileceği belirlenecek şekilde oluşturulmuştur.

Paralel kanaldan gelen giriş_çıkış komutlarının, bu komutların yürütüleceği kişisel bilgisayar çevre biriminde eşdeğer komut karşılıklarının olmaması durumunda, bu komutlar yazılım ile bir araya geldiklerinde aynı işi görecek birden çok giriş_çıkış işlemi ile gerçekleştirilmektedir. Uygulama katmanı, bu tür giriş_çıkış komutlarının işletilmesi durumunda, dönüşüm katmanına yürütülen komut için tek cevap aktaracak şekilde

oluşturulmuştur. Bu sayede, dönüşüm katmanı paralel kanal komutunun işletilmesi için gereken ayrıntılardan soyutlanmıştır.

Kişisel bilgisayar çevre birimleri ile ana bilgisayar çevre birimleri arasında fiziksel tasarımlarından kaynaklanan geometrik farklılıkların giderilmesi de uygulama katmanında gerçeklenmiştir. Özellikle, disk yapısındaki çevre birimlerinde, veri depolama kapasitelerinin yeterliliğinin yanısıra silindir, iz ve bölüm (cylinder / track / sector) uyuşmasının sağlanması da uygulama katmanında gerçeklenmektedir.

Uygulama katmanının gerçeklenmesinde, bu katmanın çalışacağı değişik merkezi işlemcilerden ve işletim sistemlerinden kaynaklanacak farklılıklardan olabildiğince az etkilenilmesi için üst seviyeli bir programlama dilinin kullanılmasına ve işletim sisteminin sağladığı çoklu iş ortamına POSIX (Portable Operating System Interface for Computer Environments) uyumlu işlem parçacığı arayüzü ile erişilmesine karar verilmiştir.

Giriş_çıkış işlemlerinin yürütülmesinde işletim sistemleri arasında cihaz sürücülerine erişimde uygulanan yerleşmiş standartlar olmadığı için, uygulama katmanının çevre birimleri ile haberleşmesi ve giriş_çıkış işlemlerini yürütmesi için gerekli olan program parçaları çoklu işlem ilgili kısımlarda olduğu gibi bir işletim sisteminden diğerine rahatça taşınabilir ve uygulanabilir olamamaktadır.

4.4.3 Uygulama katmanı ile dönüşüm katmanının bağlantısı

Uygulama katmanı ile dönüşüm katmanı arasında, giriş_çıkış birimlerinin farklılıklarını ve iç özelliklerini alttaki katmanlardan gizleyecek ve bütün giriş_çıkış birimlerine aynı programlama modeli ile erişilmesini sağlayacak bir yapı oluşturulmuştur. Uygulama katmanı ile dönüşüm katmanı arasında yürütülecek giriş_çıkış işlemine ait kontrol bilgisi aktarımının yanısıra veri aktarımı da gerçekleşmektedir. Giriş_çıkış işlemlerine ait kontrol ve sonuç bilgileri, iki katman arasında, istek-cevap (request-response) yapısında aktarılmaktadır. Dönüşüm katmanı gönderdiği isteklerde, hangi giriş_çıkış biriminin kullanılacağını, okuma, yazma, kontrol işlemlerinden hangisinin yapılmasını istediğini, varsa veri aktarım arabelleğinin yerini belirtir. Uygulama katmanından, dönüşüm

katmanına ise cevabın hangi giriş_çıkış birimi için yollandığı, giriş_çıkış biriminin durumu ve varsa veri aktarım arabelleğinin yeri bilgileri aktarılır. Şekil 4.9'de uygulama katmanı ile dönüşüm katmanı arasında haberleşmek için kullanılan yapı görülmektedir.

Giriş_çıkış birimi tanımlayıcısı	Giriş_çıkış komutları / cevapları	Giriş_çıkış komutları ve cevaplarına ait parametreler	varsa ilk arabellek adresi

Şekil 4.9 Uygulama katmanı ile dönüşüm katmanı arasında haberleşme için kullanılan yapı

Kişisel bilgisayar sisteminde, ana bilgisayar sisteminin kullanımına açılan giriş_çıkış birimlerinde oluşan sıradışı ve bu birimlerde durum bilgisi değişimine neden olan gelişmeler de uygulama katmanı tarafından dönüşüm katmanına aktarılırlar.

4.4.4 Uygulama katmanının istatistiki modeli

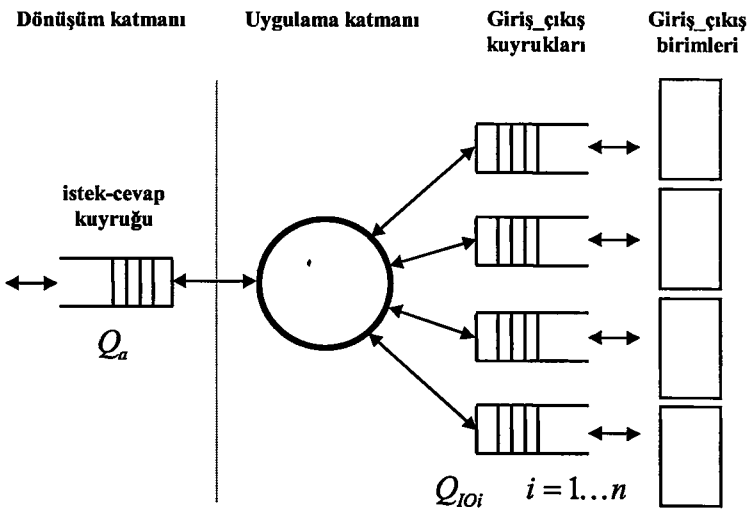
Uygulama katmanının, dönüşüm katmanı ve çevre birimleri ile ilişkisi
Şekil 4.10'da görüldüğü gibi kuyruk yapılarından yararlanılarak tariflenmiştir.

Uygulama katmanı, aynı anda birden çok istemciye hizmet veren ve birden çok kaynağı yöneten bir sunucu olarak çalışmaktadır. Uygulama katmanı ile dönüşüm katmanı arasındaki kuyruk bu iki katman arasında istek-cevap aktarımında kullanılmaktadır. Uygulama katmanının çevre birimler ile bağlantısı da çevre birimlerine ait cihaz sürücüleri ile uygulama katmanı ararsındaki kuyruklar üzerinden gerçekleştirilmektedir.

Uygulama katmanı ile dönüşüm katmanı arasındaki istek-cevap kuyruğu Q_a, uygulama katmanı ile n adet çevre birimi arasındaki giriş_çıkış kuyrukları ise,

$Q_{IOi} \quad i = 1...n$

ile ifade edilmiştir.



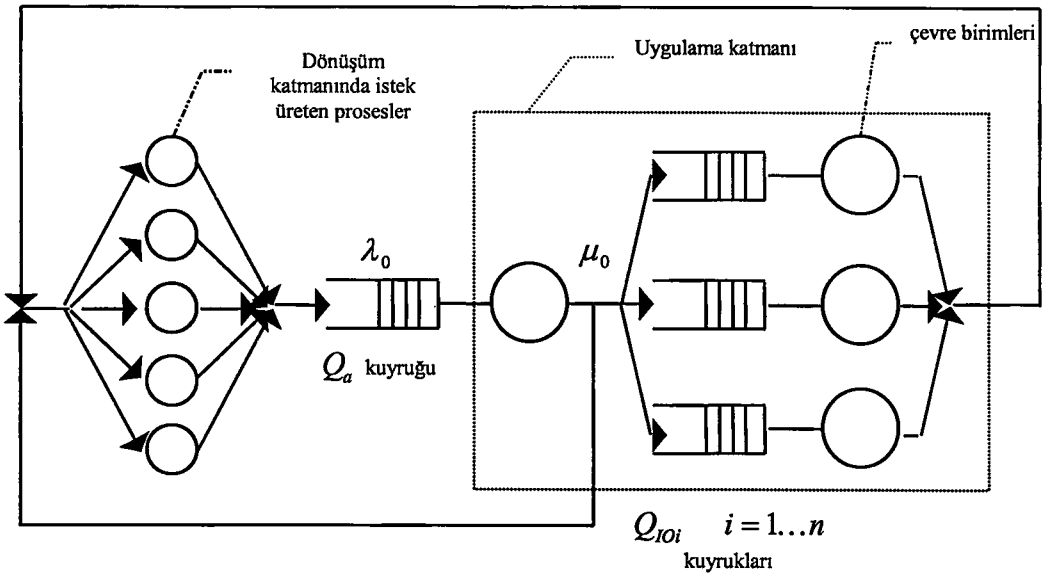
Şekil 4.10 Uygulama katmanı ile dönüşüm katmanı arasında istek-cevap modeli ve uygulama katmanında gerçekleştirilen anahtarlama yapısı

Dönüşüm katmanından Q_a kuyruğuna bir istek yollandığında bu istek ait olduğu Q_{IOi} kuyruğu üzerinden çevre birimlerinde işlenip sonuç elde edilene kadar, bu isteği üretmiş olan dönüşüm katmanı prosesi bekleme durumunda kalır. Giriş_çıkış işlemi ilgili çevre biriminde tamamlandıktan sonra, uygulama katmanından dönüşüm katmanına isteğin sonuçlandığını belirten cevap aktarılır ve bekleme durumunda olan proses çalışmaya devam eder. Katmanlar arasındaki bu ilişki Şekil 4.11'de verilmiştir.

η_0 ile Q_a kuyruğunda bekleyen istekler, $\eta_i \quad i=1...n$ ile $Q_{IOi} \quad i=1...n$ kuyruklarında bekleyen istekler gösterilecek olursa uygulama katmanında herhangi bir anda bekleyen toplam istek sayısı

$$\eta = \eta_0 + \sum_{i=1}^{n} \eta_i \qquad (4.15)$$

bağıntısı ile ifade edilir. η değeri aynı zamanda uygulama katmanındaki çoklu programlama seviyesini de göstermektedir.



Şekil 4.11 Dönüşüm katmanı ve uygulama katmanındaki kuyruklar arasındaki ilişki

Uygulama katmanına gelen isteklerin hızı λ_0, bu isteklerin Q_a kuyruğundan çıkıp ilgili çevre birimi kuyruğuna dağılım hızı da μ_0 ile gösterilecek olursa, uygulama katmanının bir isteğin hangi kuyruğa ait olduğunu belirleme ve giriş_çıkış ile ilgili ön işlemleri gerçekleştirmesini kapsayan τ_0 servis süresi

$$\tau_0 = \frac{1}{\mu_0} \qquad (4.16)$$

bağıntısı ile elde edilir. İsteklerin cihaz sürücülerine dağılımlarına ait olasılıklar

$$p_i \quad i = 1...n$$

ile gösterilirse cihaz sürücülerinin girişlerine isteklerin gelme hızı

$$\lambda_i = \mu_0 p_i \quad i = 1...n \qquad (4.17)$$

bağıntısı ile ifade edilir.

Cihaz sürücülerine ait kuyrukların incelenemesinde ise

1. her cihaz sürücüsünün bağımsız üstel hizmet süresi dağılım fonksiyonuna sahip olduğu

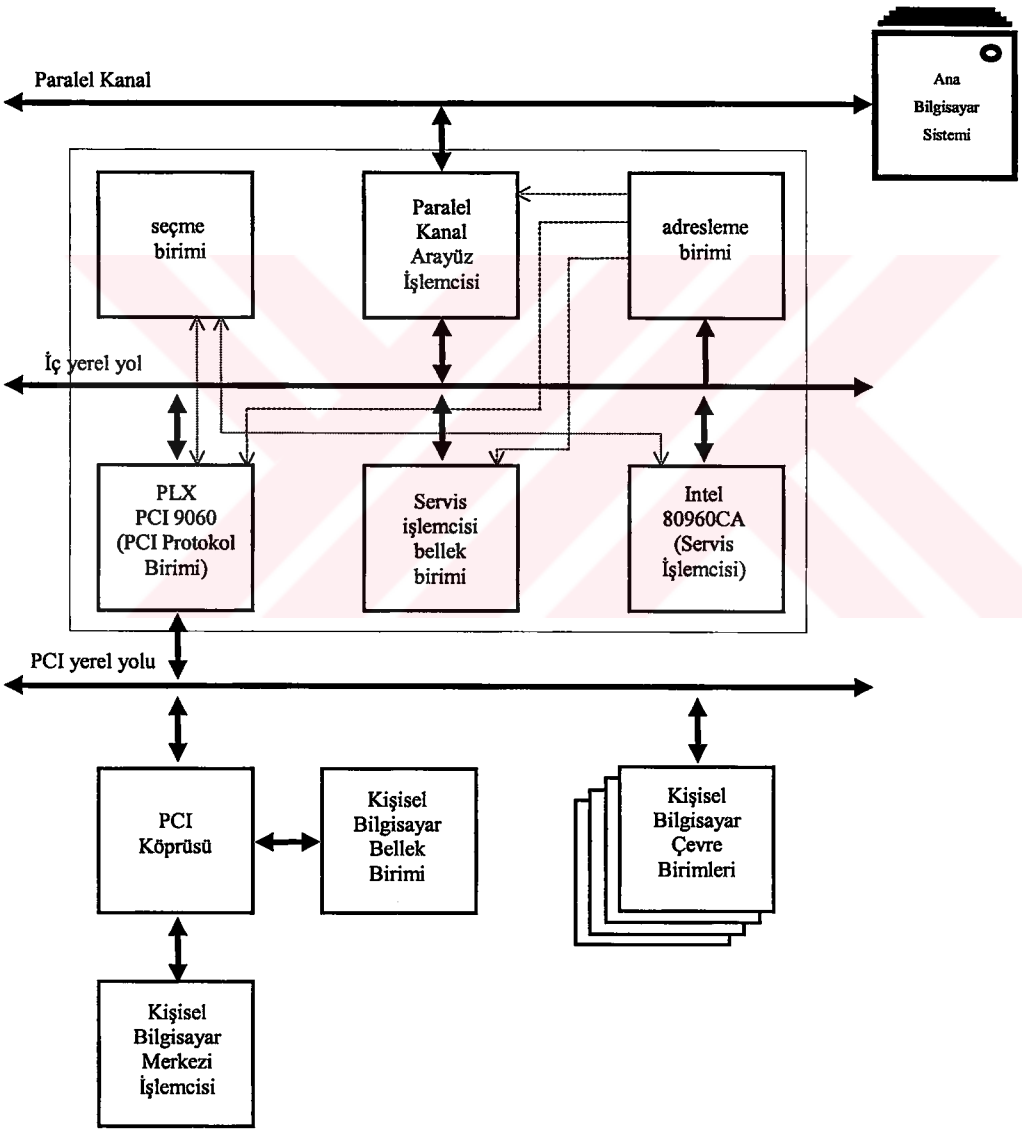
2. dönüşüm katmanından gelen isteklerin Poisson dağılımına uyduğu

3. isteklerin çevre birimlerine dağılım olasılıklarının sabit olduğu

varsayımları ile Jackson teoreminden yararlanılması ile her kuyruğun çalışması dönüşüm katmanındaki kuyruk modeline uygun değerlendirilebilir.

5. DONANIM ÖZELLİKLERİ

4. bölümde modellemesi yapılan, ana bilgisayar sistemlerindeki paralel kanal yapısı ile kişisel bilgisayar sistemlerindeki PCI yerel yolunun birleştirilmesi ve kişisel bilgisayar sistemlerindeki çevre birimlerinin ana bilgisayar sistemlerine kullandırılmasını sağlayan sistemi gerçekleştirilmek için oluşturulan donanımın yapısı Şekil 5.1'de kesikli çizgiler içerisinde görülmektedir.



Şekil 5.1 Paralel kanal yapısı ile PCI yerel yolu arasında bağlantı için gerçekleştirilen donanımın yapısı

Bu tez çalışmasında geliştirilen donanım, 32 adet adres ve veri hattı ile kontrol hatlarından oluşan iç yol (internal bus) üzerinden birbirlerine bağlanmış 5 birimden oluşturulmuştur:

1. Paralel kanala erişimi sağlayan paralel kanal protokol birimi

2. PCI yerel yoluna erişimi sağlayan PCI protokol birimi

3. Servis işlemcisi ve servis işlemcisi yerel bellek birimi

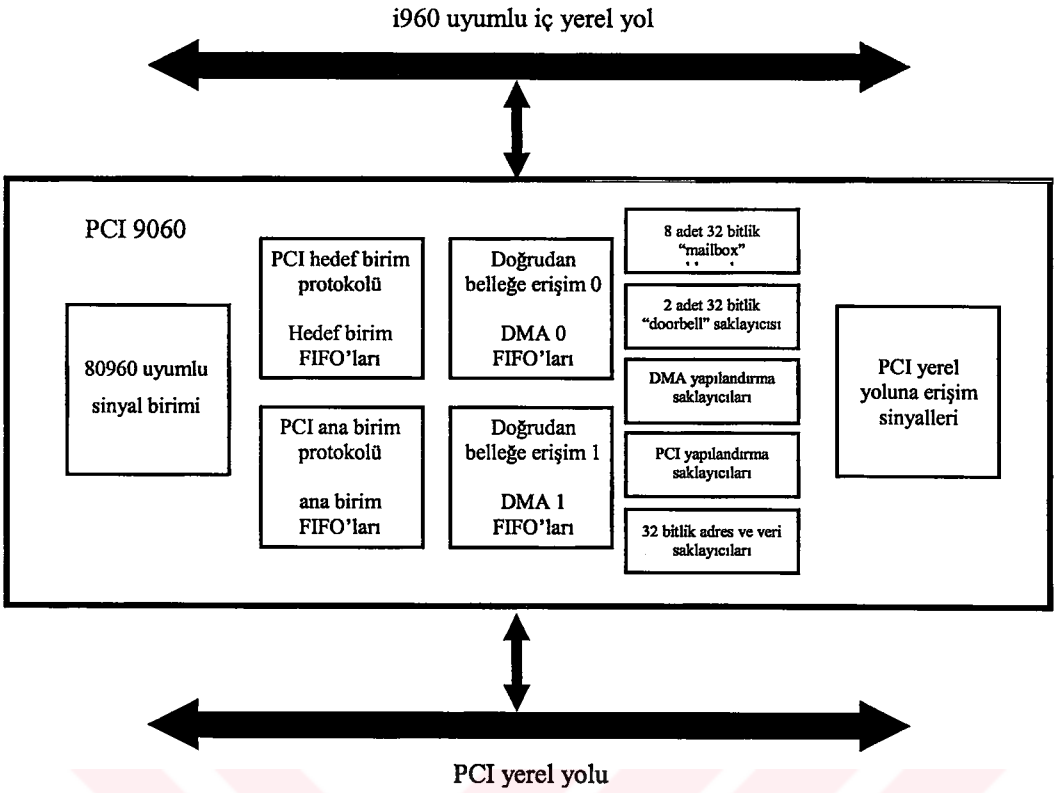
4. Birimler arasında adreslemeyi sağlayan adresleme lojiği

5. İç yola erişimi düzenleyen seçme lojiği

5.1 PCI Protokol Birimi

Gerçekleştirilen sistemde, PCI yerel yoluna erişimi sağlamak amacıyla, PCI yerel yoluna bağlantı için tasarlanmış özel amaçlı entegre devreler incelenmiş ve bu incelemenin sonunda modelde belirtilen özelliklere en çok PLX firması tarafından üretilen PCI 9060 entegre devresinin uyum sağladığı belirlenmiştir.

İç yapısı ve bağlantı özellikleri Şekil 5.2'de verilmiş olan PCI 9060 entegre devresi, arada lojik seviye dönüştücülerine gerek olmaksızın, doğrudan PCI yerel yoluna bağlanacak şekilde tasarlanmış bir birimdir. Şekil 5.2'de de görüldüğü gibi PCI 9060 entegre devresi, PCI yerel yoluna erişecek birimlerin kendisine olan bağlantısını kolaylaştırmak amacı ile Intel i960 protokolünü kullanan bir iç yerel yol yapısı da içermekte ve PCI yerel yolu ile kendi iç yerel yolu arasında köprü görevi üstelenerek, bu yollara bağlı birimlerin birbirlerine erişmelerini sağlamaktadır. PCI yerel yolu açısından değerlendirildiğinde PCI 9060, hem ana birim hem de hedef birim olarak çalışabilmektedir. PCI yerel yoluna bağlı birimler, iç yerel yola bağlı birimlere erişmek istediklerinde, PCI 9060, hedef birim olarak davranmakta, iç yerel yola bağlı birimler PCI yerel yoluna erişmek istediklerinde ana birim olarak görev yapmaktadır.



Şekil 5.2 PCI 9060 entegre devresinin iç yapısı ve bağlantı özellikleri

5.1.1 PCI yerel yolu ile fiziksel bağlantı

PCI 9060 entegre devresi, fiziksel katmanın modellenmesinde belirtilen +5V ile çalışmada, arada özel lojik seviye dönüştürücülerine ihtiyaç duyulmadan PCI yerel yoluna bağlanılabilmesi şartını da yerine getirmektedir. +3.3V ile çalışan PCI yerel yolunun kullanılması durumunda bu entegre ile bacak bağlantısının yanısıra fonksiyonel olarak eşdeğer olan PCI 9080 entegre devresinin kullanılması ve PCI yerel yolunda +3.3V, iç yerel yolda +5V çalışma geriliminin seçilmesi yeterli olmaktadır.

5.1.2 Veri genişliği ve aktarım özellikleri

PCI9060 biriminin, iki yerel yol arasında anlık veri aktarımlarında, saniyede 132 Mbyte'a varan aktarım hızları ile çalışması, PCI yerel yolunun veri aktarımlarında gereğinden fazla meşgul edilmesini ve aynı anda daha çok isteğe cevap verilebilmesini sağlamaktadır.

İç yerel yolda, PCI 9060 entegresinin, 8, 16 ve 32 bitlik veri genişliğine sahip birimler ile çalışılabilmesi, gerçekleştirilen tasarımda iç yerel yola değişik bit uzunluğunda saklayıcılara sahip birimlerin bir arada kullanılmasını da mümkün hale getirmiştir.

5.1.3 PCI yerel yolunda seçme işlemine katılım

PCI 9060, PCI yerel yoluna ana birim olarak erişim sırasında ihtiyaç duyulan seçme işlemini, protokol katmanının üstündeki katmanlara saydam bir şekilde gerçekleştirecek yapıya sahiptir. Bu sayede, üstteki katmanlar, seçme işleminde bulunulan aşamadan bağımsız olarak PCI yerel yoluna erişim isteklerini yollayabilirler. PCI 9060, iç yerel yolu, servis işlemcisi ile paylaştığı için, PCI yerel yolunda olduğu gibi, iç yerel yola erişmek istediğinde de bir seçme işlemine katılmak zorundadır. Bu seçme işlemi de PCI yerel yolundan gelen isteklere tamamen saydam gerçekleştirilmektedir.

5.1.4 Protokol birimi üzerinden parametre aktarımı

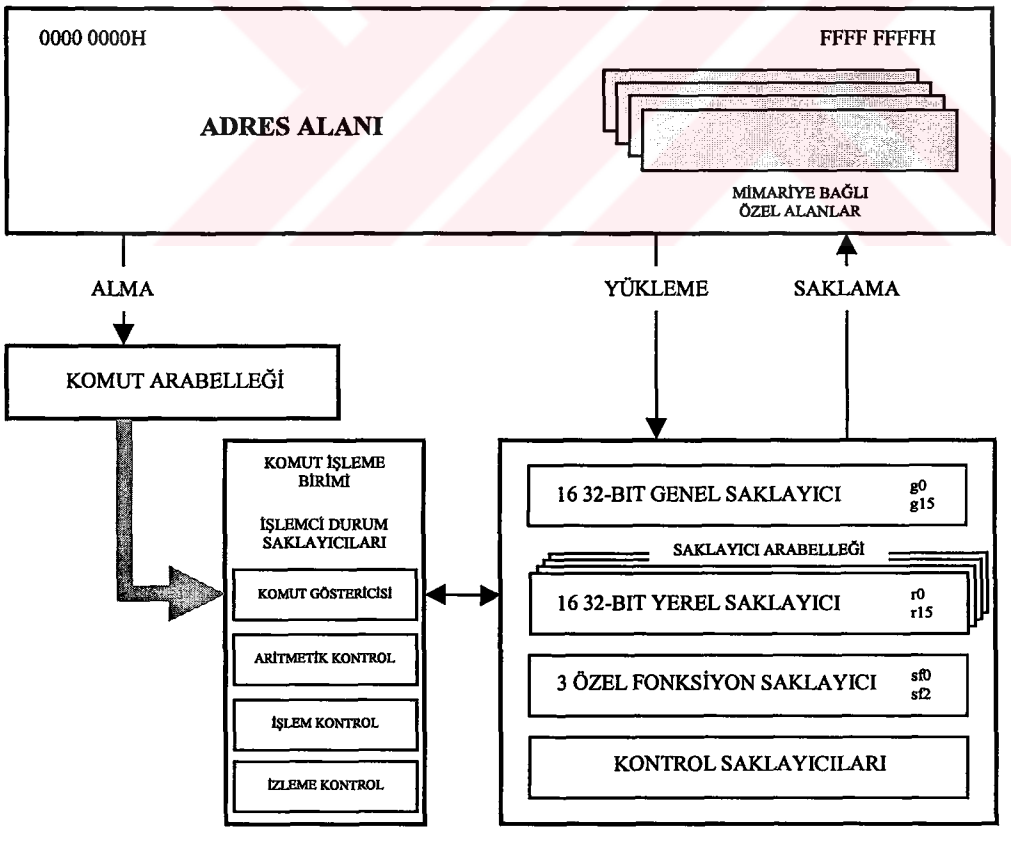
Protokol katmanında hızlı parametre aktarımı ve birimler arası senkronizasyon için öngörülen özel saklayıcılar ise PCI 9060 mimarisinde yer alan 8 adet 32-bitlik "mailbox" ve 2 adet 32-bitlik "doorbell" saklayıcıları kullanılarak gerçekleştirilmiştir. "Doorbell" saklayıcıları, PCI yerel yolundaki birimlerin iç yerel yoldaki birimleri, iç yerel yoldaki birimlerin de PCI yerel yolundaki birimleri, özel durumlardan haberdar etmeleri için gereklidirler. "Doorbell" saklayıcılarından biri PCI yerel yolundaki birimlerin, diğeri iç yerel yoldaki birimlerin yazmasına açıktır. PCI 9060 entegresi, bir taraftaki saklayıcıya yeni bir bilgi yazıldığında diğer tarafta bu durumu belirtmek için bir kesme isteği oluşturur.

Gerçekleştirilen sistemde, dönüşüm katmanı içerisinde, servis işlemcisi ile merkezi işlemci arasında istek-cevap şeklindeki haberleşme ve parametrelerin aktarımı, "mailbox" ve "doorbell" saklayıcıları kullanılarak sağlanmıştır. "Mailbox" saklayıcıları, 32 byte'ı geçmeyen isteklerin, cevapların veya parametrelerin karşı tarafa aktarılması, "doorbell" saklayıcıları ise karşı tarafı, "mailbox" saklayıcılarında bekleyen bilgilerden haberdar

etmek için kullanılmıştır. "Doorbell" saklayıcılarına yeni veri yazılması ile kesme isteği oluşturulmuş ve ilgili işlemcinin yapılan veri aktarımını fark etmesi sağlanmıştır.

5.2 Servis İşlemcisi

Paralel kanala erişilmesi, seçme isteklerine cevap verilmesi, veri aktarımının yapılması gibi paralel kanal ile ilgili işlemlerin protokol katmanında, servis işlemcisinin denetiminde yazılım kontrollü donanım ile gerçekleştirilmesinin yanısıra dönüşüm katmanına ait fonksiyonların da servis işlemcisi tarafından yürütüleceği göz önüne alınarak, hem kontrol uygulamalarına uygun hem de yüksek performanslı bir işlemcinin, servis işlemcisi olarak seçilmesine karar verilmiştir. Servis işlemcisinin seçilmesinde, PCI protokol birimi olarak PCI 9060 entegre devresinin seçilmiş olması ve bu entegre devrenin iç yerel yolunda i960 yol protokolünü desteklemesi de önemli rol oynamıştır. Yapılan araştırma sonunda, servis işlemcisi olarak 32-bitlik Intel 80960CA işlemcisi seçilmiştir (Şekil 5.3).



Şekil 5.3 80960 işlemcisinin iç yapısı ve programlama modeli (Intel, 1994)

5.2.1 Yol protokolü ve önemli mimari özellikleri

PCI 9060 entegresinin iç yerel yol yapısı ile bu yol üzerinde kullandığı protokolün 80960CA işlemcisinin yol yapısı ile aynı olması bu iki birimin birbirleri ile olan bağlantısının en az sayıda çevre elemanı kullanılarak yapılmasını sağlamıştır.

32 bitlik adres ve veri yolu yapısı, aynı anda ortalama 2, en çok 3 komutu birden işletebilmesi, yükleme ve saklama (load / store) türünde temel işlemlere dayalı komut kümesi, 32 adet 32 bitlik saklayıcıyı içermesi ve 64 bit uzunluğunda bit alanları üzerinde bit işlemleri yapabilmesi, 80960CA işlemcisinin servis işlemcisi olarak seçilmesinde önem taşımıştır.

80960CA işlemcisinin, üzerinde bulunan 1 Kbyte'lık önbelleğe istenilen bir program parçasının yüklenebilmesine ve önbellekten uzaklaştırılmasının engellenmesine izin vermesi, özellikle kesme servis programlarının, kesme isteklerine cevap süresini önemli ölçüde iyileştirmektedir. Böylece kesme isteği oluştuğunda, işlemci dışına erişim yapılmasına ihtiyaç olmaksızın, kesme servis programına dallanılabilmekte ve gelen kesme isteğine hızlı bir şekilde servis verilebilmektedir.

5.2.2 Prosesler arasında donanım ile geçişin sağlanması

İşletim sistemlerindeki proses mantığı, 80960CA işlemcisi tarafından donanım olarak tanımlı olduğu için çoklu işlem (multiprogramming) ortamlarında, bir prosesten diğerine geçilirken aktif prosese ait saklayıcıların içeriklerinin saklanması ve yeni prosese ait saklayıcıların içeriklerinin işlemciye yüklenmesi işlemi, 80960CA işlemcisi tarafından donanım olarak gerçekleştirilmektedir. Donanım, en çok 15 proses arasında geçişlere kullanıcı müdahalesine ihtiyaç olmadan izin vermekte, proses sayısının artması durumunda yazılım olarak saklayıcı içeriklerinin saklanması gerekmektedir. Bu durum göz önünde bulundurularak, gerçekleştirilen sistemde, protokol ve dönüşüm katmanlarına ait toplam proses sayısı 15 ile sınırlandırılmıştır.

5.2.3 80960CA işlemcisinin performansının sisteme etkisi

80960CA işlemcisinin seçilmesinde paralel kanal ile gerçekleştirilen işlemlerde uyulması gereken zaman kısıtları da etkili olmuştur. Paralel kanalda seçme, veri aktarımı ve sonlandırma işlemleri sırasında sinyallerin aktif kalma ve geri çekilme sürelerinde belirtilen sınırların içerisinde kalınması, hem kanalın hem de kontrol biriminin performansını doğrudan etkileyen unsurlardır.

Örneğin, paralel kanalda yürütülen seçme işlemlerinde, adreslenen giriş_çıkış biriminin servis işlemcisinde kayıtlı adresler içerisinde olmadığının belirlenmesi ve isteğin bir sonraki kontrol birimine aktarılmasında oluşacak gecikme süresi 600 ns ile 1800 ns arasında olacak şekilde belirlenmiştir. Seçme işlemini başlatan Select Out sinyalinin, giriş_çıkış birimine ait adres bilgisinin BusOut yolunda olduğunu belirten Address Out sinyalinin aktif olmasından en az 400 ns sonra aktif olması gerektiği göz önüne alınacak olursa, servis işlemcisinin Address Out sinyalinin aktif olduğunu belirlemesinden sonra bu seçme isteğine cevap vermek için en az 1000 ns en çok 2200 ns süresi olduğu ortaya çıkar.

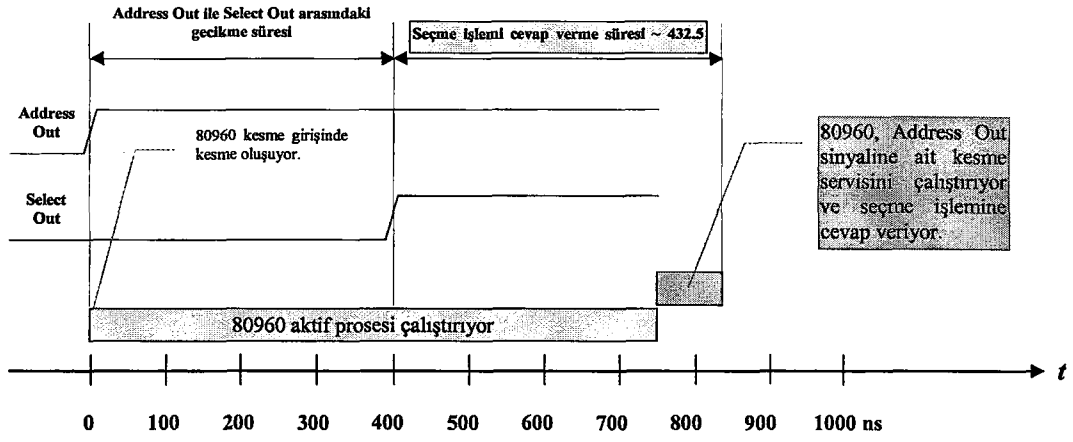
Gerçekleştirilen sistemde, bu tür zamanlamalarda performansı arttırmak amacıyla minimum değerler ile çalışılması ilkesine uyulduğu için seçme işleminde kiritik zaman olarak 1000 ns kabul edilmiş ve buna göre tasarım yapılmıştır. Paralel kanalda olan sinyal değişikliklerinin farkına varmak için örnekleme (polling) veya kesme yönteminden yararlanılabilir. Servis işlemcisi dönüşüm katmanına da hizmet vereceğine göre çoklu işlem ortamından yararlanılması ve paralel kanala ait sinyallerin kesme yöntemi ile izlenmesi gerekmektedir. 80960CA işlemcisi, 750 ns lik kesme gecikmesi süresi ile 1000 ns'lik kiritik zaman sınırları içine düşmektedir. İşlemcinin 33 MHz ile çalıştığı ve her sistem saatinde ortalama iki komut yürüttüğü göz önüne alınacak olursa geriye kalan 250 ns'lik süre yaklaşık 15 komut için yeterli olmaktadır. Bu durumda Address Out sinyalinin, 80960CA işlemcisinde bulunan yüksek öncelikli kesme girişlerinden birine bağlanması ve bu kesme girişine servis veren özel bir kesme servis programının çalıştırılması istenen cevap süresini sağlayacaktır. Performansı arttırmak amacıyla bu servis programının ön bellek içerisinde kilitlenmesi yöntemi de uygulanmıştır.

5.2.4 80960 işlemcisinin performansının algoritmik yöntemlerle arttırılması

Sistemin gerçekleştirilmesinde donanımın sağladığı özelliklerden daha iyi yararlanmak ve sistemin performansını arttırmak amacıyla yazılım yönünden optimizasyon ve özel algoritmaların uygulanması yoluna gidilmiştir. Bu tez çalışması kapsamında paralel kanaldan adresi aktarılan giriş_çıkış biriminin kayıtlı olup olmadığını 15 komutu aşmayacak şekilde belirlemek amacıyla özel bir algoritma geliştirilmiştir.

Geliştirilen algoritma, 80960CA işlemcisinin bit alanları üzerinde doğrudan işlem yapabilmesi özelliğinden yararlanmaktadır. Her kontrol birimine bir kontrol birimi taban adresi verilmiş ve kontrol biriminin hangi giriş_çıkış birimlerine servis verdiği 32 bitlik bit alanlarına yerleştirilmiştir. Bu bit alanları en sağda 0. giriş_çıkış birimi, en solda ise 31. giriş_çıkış birimi ifade edilecek şekilde düzenlemişlerdir. Bir bitin 0 değerine sahip olması, o giriş_çıkış birimine servis verilmediğini, 1 olması ise o giriş_çıkış birimine servis verildiğini belirtmek için kullanılmıştır.

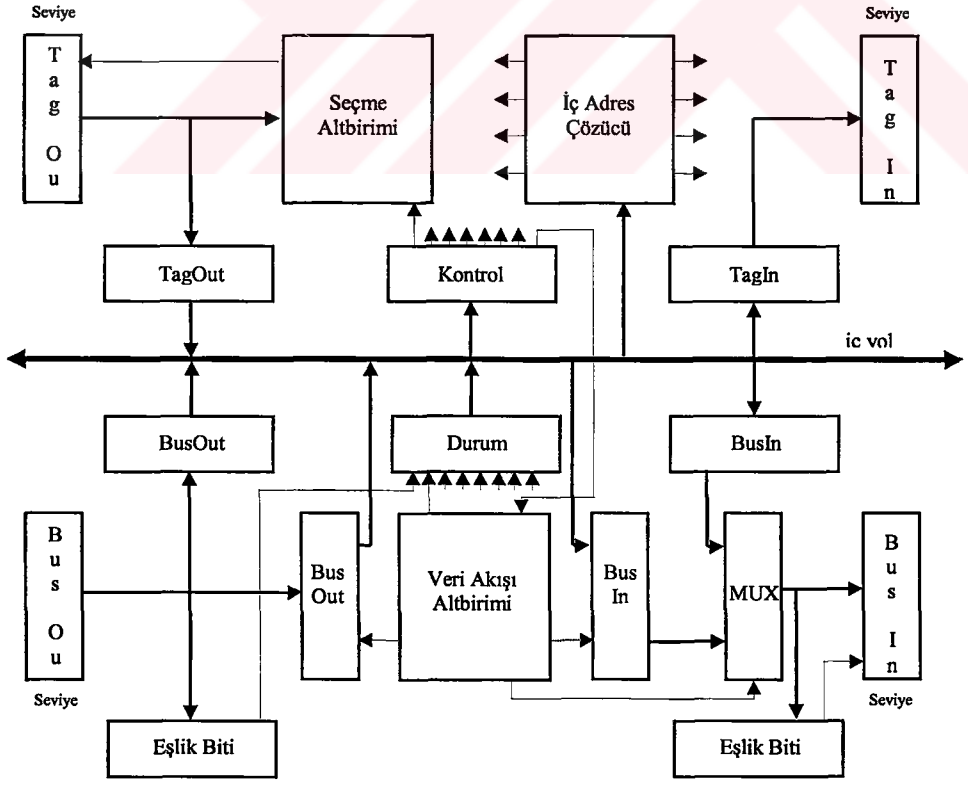
Seçme isteğinde gelen adresten kontrol biriminin temel adresi bir "ve" işlemi ile elde edilmekte ve hangi kontrol birimine ait bilgiler ile işlem yapılacağı belirlenmektedir. Taban adres çıkarıldıktan sonra geriye kalan adres, giriş_çıkış biriminin adresini ifade ettiği için, bu değer "Scan for Bit" komutuna aktarılmakta ve kontrol birimine ait 32 bitlik bit alanında o bitin değeri elde edilmektedir. Bu yönteme göre bir giriş_çıkış birimine servis verilip verilmeyeceği yaklaşık 5 komut ile belirlenmektedir. Her sistem saatinde ortalama 2 komut yürütüldüğü bilindiğine göre bu işlem yaklaşık 2.5 sistem saatinde tamamlanmaktadır. Bu durumda Address Out sinyalinin aktif hale gelmesi referans alındığında (750 + 82.5) ns de cevap üretilebilmektedir (Şekil 5.4).



Şekil 5.4 80960'ın Address Out kesmesine cevap verme süreci

5.3 Paralel Kanal Protokol Birimi

Bu tez çalışması kapsamında, paralel kanala erişimi sağlamak amacıyla özel bir devre tasarlanmış ve gerçeklenmiştir. "**Paralel Channel Interface Processor**" (**PCIP**) olarak isimlendirilen bu devrenin blok şeması Şekil 5.5'de görülmektedir.



Şekil 5.5 Gerçeklenen PCIP biriminin blok şeması

PCIP biriminin tasarımında, TTL uyumlu ayrık lojik elemanları ve programlanabilir lojik birimleri kullanılmıştır. PCIP birimine PCI yerel yolundan veya iç yerel yoldan yapılacak erişimlerde, bekleme çevrimi oluşmaması ve mümkün olan en düşük çevrim süresi ile erişimin tamamlanabilmesi için, tasarım düşük yayılım gecikmeli, yüksek hızlı lojik aileleri temel alınarak yapılmıştır. Bu sayede PCIP biriminde sistem performansını etkileyen saklayıcılara, 33 MHz'de çalışan PCI yerel yolundan veya iç yerel yoldan gelen isteklerde bekleme çevrimine ihtiyaç olmadan erişilmesi sağlanmıştır.

5.3.1 Bus arayüzü altbirimi

PCIP birimi, paralel kanalın BusOut ve BusIn yollarına Bus arayüzü altbirimi ile bağlıdır. BusOut yolundan gelen sinyaller, seviye dönüştürücülerinden geçirilip TTL uyumlu hale getirildikten sonra 8 bitlik BusOut saklayıcısı üzerinden iç yola aktarılırlar. İç yola bağlı 8 bitlik BusIn saklayıcısının çıkışı da seviye dönüştürücüleri ile paralel kanala uygun hale getirilir.

Paralel kanal yapısına uygun olarak BusOut yolunda gelen verilerin tek eşlik kontrolü yapıldığı gibi BusIn yoluna çıkarılan veriler için de tek eşlik değeri oluşturulur.

Servis işlemcisinden bağımsız veri akışı yöntemi ile veri aktarımını sağlayabilmek için BusOut ve BusIn yollarına "ilk giren ilk çıkar" prensibine uygun çalışan 8 Kbyte kapasitesinde ara bellek birimleri de yerleştirilmiştir.

5.3.2 Tag arayüzü altbirimi

Tag arayüz altbirimi ile, PCIP biriminin paralel kanalın TagOut ve TagIn yollarına bağlantısı sağlanmıştır. Bu altbirimde de paralel kanal ile PCIP arasında lojik seviye dönüşümünü sağlayan seviye dönüştürücüler bulunmaktadır. 8 bitlik TagOut saklayıcı üzerinden herhangi bir anda TagOut yolunun değeri okunabildiği gibi, TagIn yoluna çıkarılacak veriler 8 bitlik TagIn saklayıcısına yazılırlar.

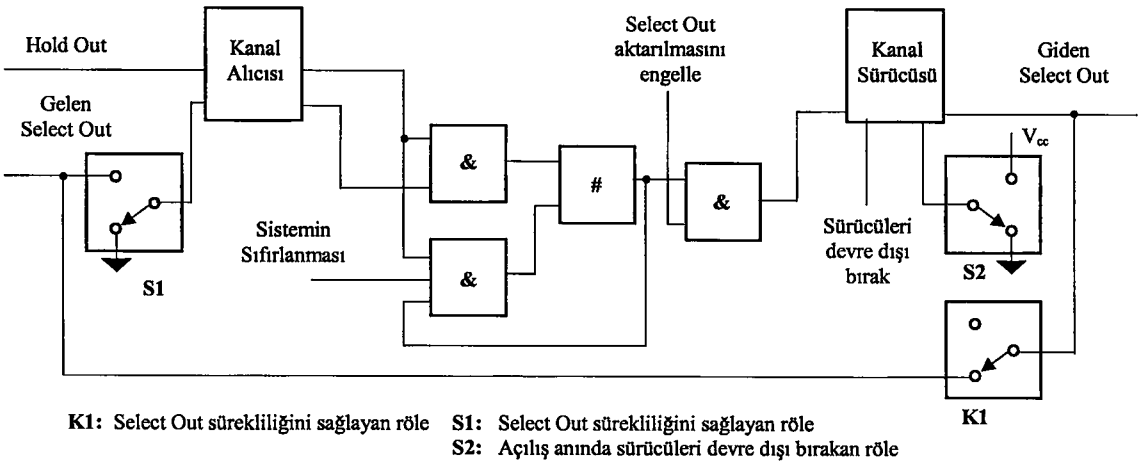
Veri akışı yöntemini desteklemek amacıyla gerçekleştirilen kontrol biriminin, TagOut yolundan gelen Service In ve Data In sinyalleri ile TagIn yoluna çıkarılan Service Out ve Data Out sinyallerine, TagOut ve TagIn saklayıcılarına ihtiyaç duymadan erişebilmesi sağlanmıştır.

5.3.3 Seçme işlemi altbirimi

Seçme altbirimi, PCIP biriminin paralel kanalda yürütülen seçme işlemlerine katılabilmesini sağlamak amacıyla gerçekleştirilmiştir. Seçme altbirimi, sistemin açılış ve kapanış anlarında Select Out ve Select In sinyallerinin elektriksel sürerkliliğini, paralel kanal kurallarına uygun sağlayacak şekilde tasarlanmıştır. Bu tasarımda sinyallerin sürekliliği, sistem kapalı iken, mekanik olarak, sistem açılıp kararlı duruma geçtikten sonra elektronik olarak sağlanmıştır.

PCIP birimi, paralel kanaldan gelen giriş_çıkış birimi adresinin kendisinde kayıtlı olduğunu ve servis verilmesi gerektiğini belirlediğinde, seçme altbirimi ile Select Out sinyalinin diğer kontrol birimlerine aktarılmasını engeller.

Seçme altbiriminde, elektriksel sürekliliğin sağlanması ve Select out sinyalinin aktarılmasını engellemek için oluşturulan yapının prensip şeması Şekil 5.6'da görülmektedir.



Şekil 5.6 Seçme altbiriminin prensip şeması

5.3.4 Veri akışı altbirimi

Veri akışı altbirimi, servis işlemcisinden bağımsız olarak, paralel kanal ile veri akışı yöntemine göre veri aktarımını yürütmek için gerçeklenmiştir.

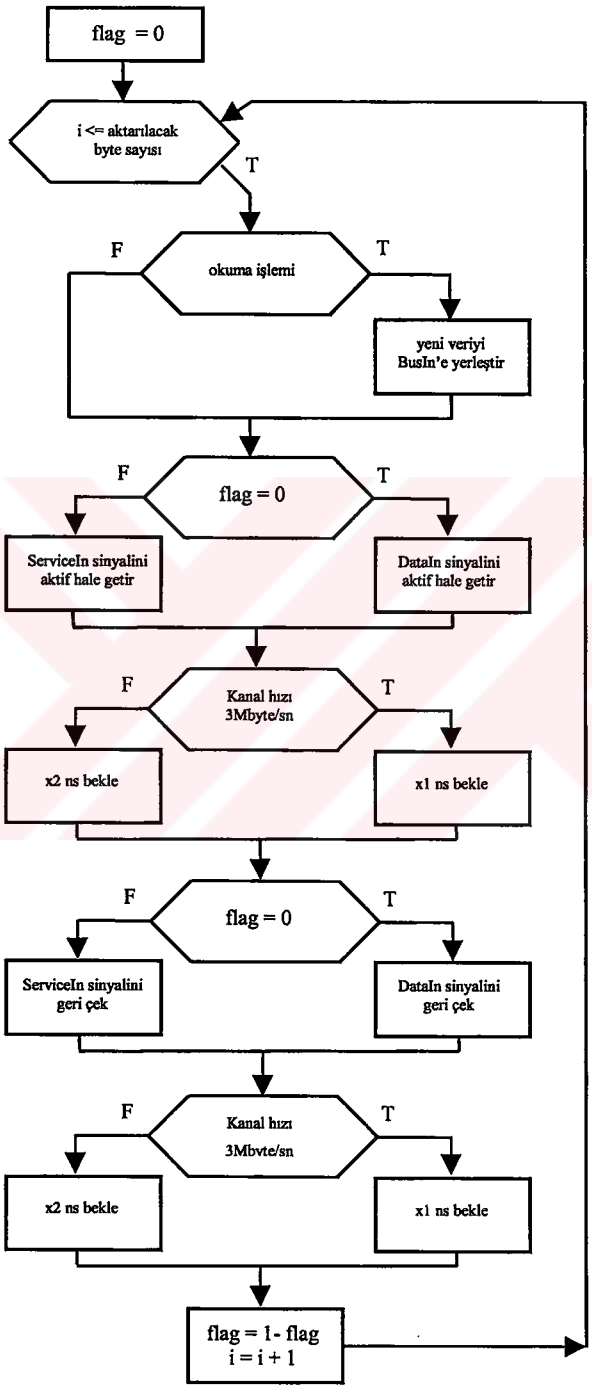
Bu alt birimin gerçeklenmesinde programlanabilir lojik birimlerinden yararlanılmıştır. Programlanabilir lojik birimlerinden yararlanılması, veri akışı yönteminin yazılım ile modellenmesini, akış diyagramının oluşturulmasını ve yazılım olarak çalışma testlerinin yapılmasını mümkün kılmıştır.

Veri akışı altbirimi kendi içerisinde istek gönderen ve cevapları kontrol eden 2 bölümden oluşmaktadır. İstek gönderen bölüm, okuma komutlarında, kanala bilgi yollandığını, yazma komutlarında ise kanaldan bilgi beklendiğini belirtmek için Service In ve Data In sinyallerinin aktif hale getirilip geri çekilmesinden sorumludur. Cevapları kontrol eden bölüm, kanalın gönderilen isteklere tanımlı süreler içerisinde cevap verip vermediğini Service Out ve Data Out sinyallerini kontrol ederek izler. Yazma komutlarında, Service Out veya Data Out sinyallerinin aktif hale geldiği anlarda BusOut üzerinden gönderilen yeni verinin okunması da cevapları kontrol eden bölüm tarafından gerçekleştirilir.

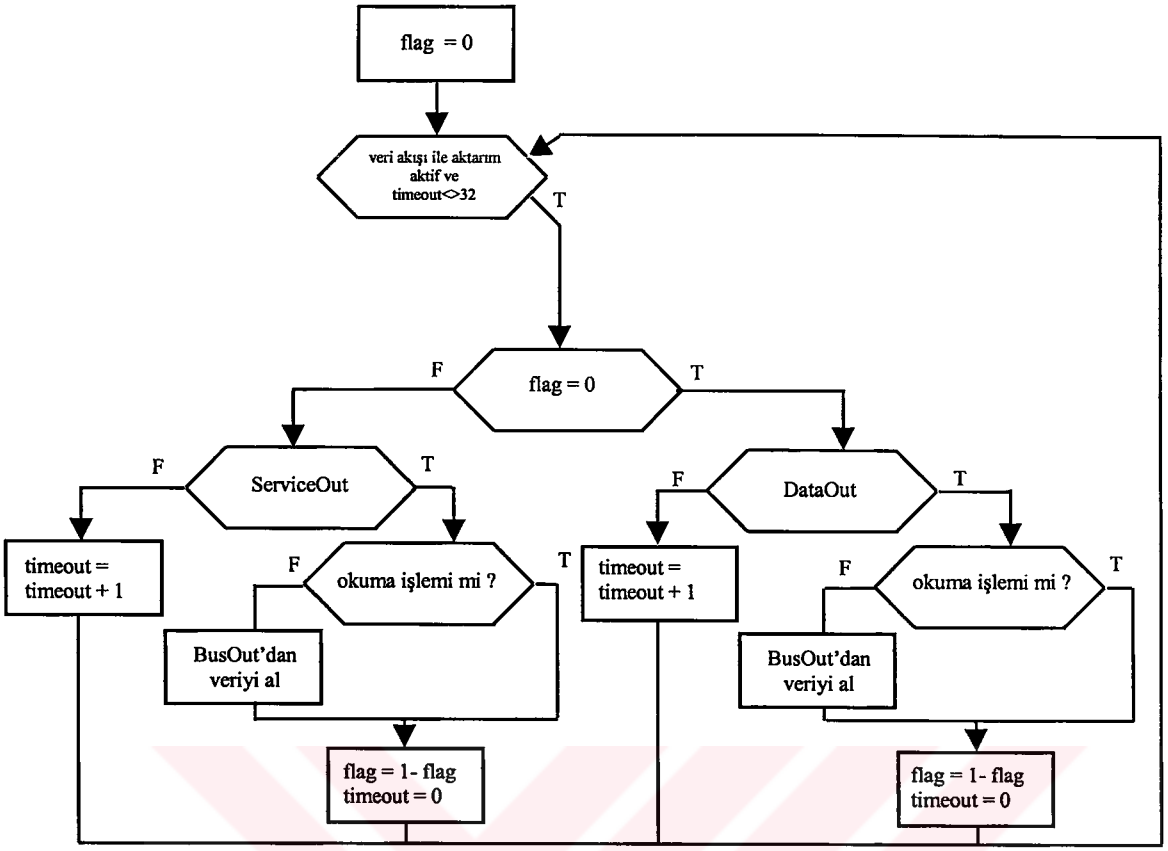
Veri akışı yöntemi ile yürütülen veri aktarımlarında, servis işlemcisini ve iç yerel yolu gereksiz yere meşgul etmemek ve PCIP biriminin performansını arttırmak amacıyla, kanal ile bağlantıda 2 ara bellekten yararlanılmıştır. Her biri 8 Kbyte kapasitesinde olan bu ara bellek birimleri, okuma ve yazma yönünde PCIP ile kanal arasında aktarılacak verinin geçici olarak tutulması yoluyla yeni veriye ihtiyaç olduğunda veya kanaldan yeni veri geldiğinde iç yerel yola erişim yapılmadan isteğin karşılanmasını sağlamak amacıyla uygulanmışlardır.

Veri akışı altbirimi, verinin aktarım yönünü, aktarılacak verinin byte cinsinden uzunluğunu ve aktarımın başlatılıp durudurulması bilgisini tutan iç saklayıcılara sahiptir. Servis işlemcisi bu saklayıcılar ile veri akışı altbirimin programlanmasını ve istenilen veri akışını gerçekleştirmesini sağlar.

Veri akışı altbiriminin, istek gönderen ve cevapları kontrol eden bölümlerine ait akış diyagramları, Şekil 5.7'de ve Şekil 5.8'de verilmiştir. Bu akış diyagramları ilgili bölümün çalışma prensibini göstermeye yönelik çizildikleri için, gerçekleştirilen sistemde kullanılan tüm sinyallere yer verilmemiştir.



Şekil 5.7 Veri akışı altbiriminin kanala istek gönderen bölümünün çalışmasını gösteren akış diyagramı



Şekil 5.8 Veri akışı altbiriminin kanaldan gelen cevapları kontrol eden bölümünün çalışmasını gösteren akış diyagramı

5.3.5 PCIP kontrol saklayıcısı

Kontrol saklayıcısı, PCIP içerisinde yer alan değişik birimlerin kontrol edilmesi için gerekli sinyalleri içermektedir. Kontrol saklayıcısından çıkan sinyaller, seçme altbirimini, veri akışı altbirimini, kullanılacak veri aktarımı protokolünü ve benzeri işlemleri kontrol etmek amacı ile kullanılmışlardır.

5.3.6 PCIP durum saklayıcısı

PCIP durum saklayıcısı, PCIP birimi içerisinde gerçekleştirilen işlemlerin sonuçlarına ait bilgileri tutmak için gerçeklenmiştir. Veri aktarımında oluşacak hatalar, veri akışı sırasında

yeni veriye ihtiyaç olduğunun belirtilmesi veya kullanılan arabelleğin dolduğunun belirtilmesi gibi durumların rapor edilmesi, durum saklayıcısında bulunan bitler aracılığı ile sağlanmıştır.

5.4 Paralel Kanal Protokol Biriminin Gerçekleştirilmesinde Programlanabilir Donanım Kullanılması

Paralel Kanal Protokol Birimi'nin gerçekleştirilmesinde TTL uyumlu ayrık donanım ve programlanabilir donanım birimlerinden yararlanılmıştır.

PCIP birimi, paralel kanala erişim hakkının elde edilmesi (selection / arbitration), paralel kanal üzerinden veri aktarımının başlatılması, sürdürülmesi ve sonlandırılması için gerekli protokol özelliklerini içermesinin yanısıra paralel kanal ile fiziksel bağlantıyı da sağlamaktadır. Paralel kanal ile fiziksel bağlantının sağlanması için TTL lojik seviyeleri ile paralel kanalda geçerli olan lojik seviyeleri arasında dönüşüm yapılması gereklidir. Bu dönüşüm için gerekli olan donanım birimlerinin fonksiyonu, programlanabilir donanım birimleri ile gerçekleştirilemeyeceği için, fiziksel bağlantının sağlanmasında ayrık donanımdan yararlanılmıştır. Fiziksel bağlantı dışındaki fonksiyonların gerçekleştirilmesinde ise programlanabilir donanım birimleri kullanılmıştır.

5.4.1 Programlanabilir donanım birimleri

Programlanabilir donanım birimlerinde, yapılandırmaları değiştirilebilir lojik ve flip-flop kaynakları bulunmaktadır. Lojik ve flip-flop kaynaklarının yapılandırmalarının değiştirilebilmesi ve aralarındaki bağlantının programlanabilir olması, birden fazla ayrık donanım elemanı kullanılarak oluşturulan tasarımların, kapasitesine bağlı olarak tek bir programlanabilir donanım birimi ile gerçekleştirilebilmesini sağlamaktadır.

Programlanabilir donanım birimlerinde kaynakların yapılandırması ve aralarındaki bağlantının şekli, bellek hücreleri (memory cells) ile değiştirilmektedir. Bellek hücreleri, içeriği bir kez belirlenebilir sigorta (fuse) yapısında veya yeniden programlanabilir,

EEPROM, FLASH yapısında olabilmektedirler. İç yapılarına ve içerdikleri lojik ve flip-flop kaynaklarına bağlı olarak programlanabilir donanım birimleri dört temel sınıfa ayrılmaktadırlar: (OptiMagic, 1999)

1. Basit Programlanabilir Donanım Birimleri (SPLD, Simple Programmable Logic Devices)

2. Karmaşık Programlanabilir Donanım Birimleri (CPLD, Complex Programmable Logic Devices)

3. Alan Programlanabilir Kapı Dizileri (FPGA, Field Programmable Gate Arrays)

4. Alan Programlanabilir Arabağlaşım (FPIC, Field Programmable InterConnect)

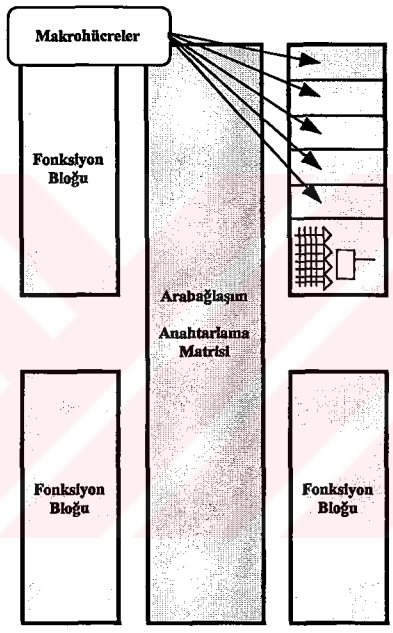
PCIP biriminin gerçekleştirilmesinde, SPLD ve CPLD sınıfı programlanabilir donanım birimlerinden yararlanılmıştır.

5.4.1.1 SPLD programlanabilir donanım birimleri

SPLD birimleri, genellikle 4 ile 22 arasında değişen sayıda makrohücre (macrocell) içermektedirler ve bir veya birden fazla 7400 serisi TTL entegre biriminin programlanabilir donanım ile gerçekleştirilmesinde kullanılmaktadırlar. SPLD'yi oluşturan makrohücreler arasında tam bir arabağlaşım bulunmaktadır ve bu arabağlaşım sigorta veya bellek hücresi teknolojileri kullanılarak gerekli bağlantılar oluşacak şekilde programlanabilmektedir. Gerçekleştirilen tez çalışmasında SPLD sınıfı programlanabilir donanım birimlerinden, 'PCIP kontrol saklayıcısı', 'PCIP durum saklayıcısı' ve 'Seçme İşlemi Altbirimi' gibi kısımların oluşturulmasında yararlanılmıştır.

5.4.1.2 CPLD programlanabilir donanım birimleri

CPLD birimleri, SPLD programlanabilir donanım birimleri ile aynı temel yapıya sahiptirler ve genellikle 2 ile 64 arasında değişen sayıda SPLD biriminden oluşmaktadırlar. CPLD'yi oluşturan SPLD birimleri, daha genel amaçlı fonksiyonel bloklar oluşturmak amacıyla 8 ile 16'lık gruplar halinde birleştirilmişlerdir. Bir fonksiyonel blok içerisinde yer alan makrohücreler arasında tam arabağlaşım olmasına karşın, fonksiyonel bloklar arasındaki arabağlaşımın derecesi değişim göstermektedir. Şekil 5.9'da CPLD programlanabilir donanım birimlerine ait genel iç yapı görülmektedir.



Şekil 5.9 CPLD programlanabilir donanım birimlerinin genel iç yapısı (OptiMagic, 1999)

Tasarlanan PCIP birimine ait "Bus Arayüzü", "Tag Arayüzü" ve "Veri Akışı" altbirimlerinin gerçekleştirilmesinde, uçtan uca (pin-to-pin) sağladıkları düşük yayılım gecikmesi (propagation delay) ve içerdikleri lojik ve flip-flop kaynakları sayesinde kombinasyonel lojik ve saklayıcı temelli durum makinası içeren tasarımlara uygunlukları göz önüne alınarak CPLD birimlerinin kullanılmasına karar verilmiştir.

5.4.2 Programlanabilir donanım birimi kullanımının sağladığı avantajlar

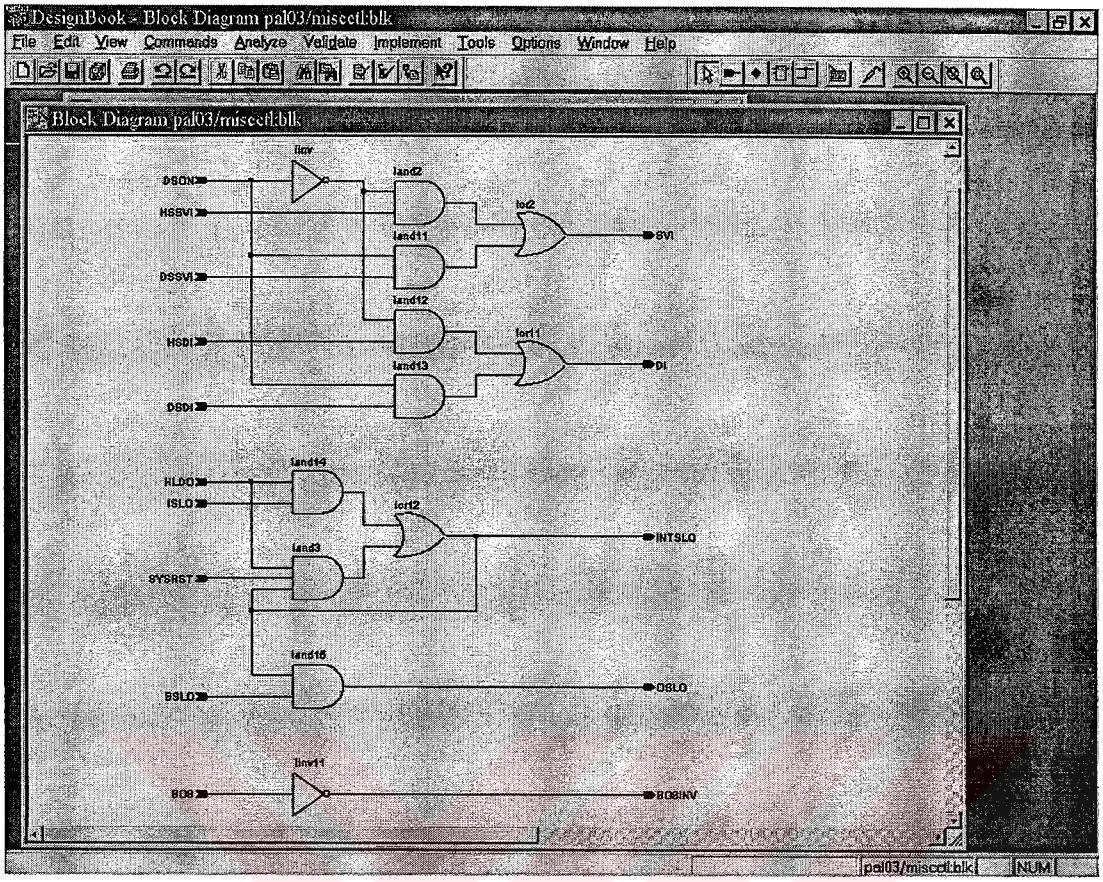
Donanım tasarımında ve gerçeklenmesinde standart ayrık donanım birimleri yerine programlanabilir donanım birimlerinin kullanılması, tasarım, üretim ve test aşamalarında sağladığı avantajlar ile giderek önem kazanmaktadır. Tablo 5.1'de, donanım tasarımında ve gerçeklenmesinde önemli olan kriterlerin ayrık donanım, programlanabilir donanım ve ASIC (Application Specific Integrated Circuit – Uygulamaya Yönelik Entegre Devre) için karşılaştırması görülmektedir.

Tablo 5.1 Ayrık donanım, programlanabilir donanım ve ASIC'in karşılaştırılması (Altera)

Kriter	Ayrık Donanım	Programlanabilir Donanım	ASIC
Hız	▒	■	■
Yoğunluk	□	■	■
Maliyet	□	■	■
Tasarım zamanı	▒	■	▒
Prototip ve simülasyon zamanı	□	■	▒
Üretim zamanı	▒	■	□
Kullanım kolaylığı	▒	■	□
Değişikliklere uygunluk	▒	■	▒

■ iyi ▒ orta □ kötü

Bu tez çalışması kapsamında tasarlanan ve gerçekleştirilen PCIP birimine ait "Seçme Altbirimi"nin (Şekil 5.10) gerçeklenmesinde ayrık donanım ve programlanabilir donanım kullanılması durumlarında gerekli kaynaklar Tablo 5.2'de görülmektedir. Bu kaynakların hız, yoğunluk(alan) ve maliyetlerinin karşılaştırılması Tablo 5.3'deki gibi elde edilmiştir.



Şekil 5.10 Seçme altbiriminin temel lojik fonksiyonlardan oluşan devre şeması

Tablo 5.2 Seçme altibiriminin gerçeklenmesinde gerekli olan ayrık ve programlanabilir donanım kaynakları

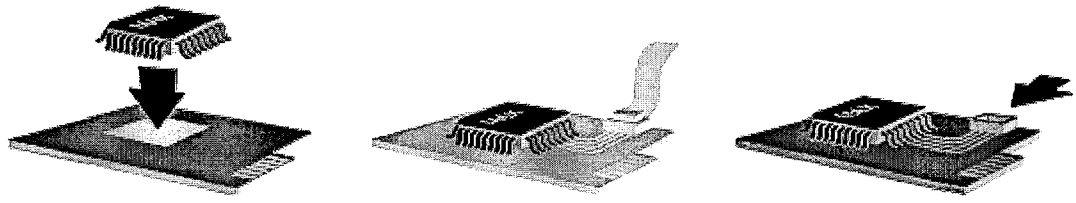
Gereken Kaynaklar	Ayrık Donanım	Programlanabilir Donanım
7 adet 'VE' kapısı (1 tanesi 3 girişli)	2 adet 74F08	
3 adet 'VEYA' kapısı	1 adet 74F32	1 adet PALCE16V8
2 adet 'DEĞİL' kapısı	1 adet 74F14	

Tablo 5.3 Seçme altbiriminin gerçeklenmesinde kullanılan ayrık ve programlanabilir donanım kaynaklarının karşılaştırılması

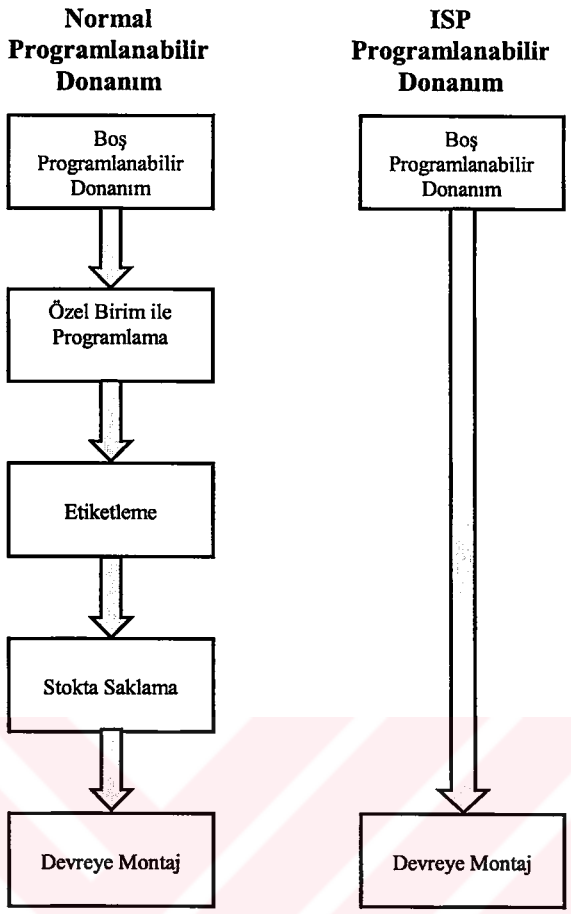
	Ayrık Donanım	Programlanabilir Donanım
Hız	15 ns	5 ns
Yoğunluk – Alan	%100 (74F08) %75 (74F32) 4.64 cm^2 %33 (74F14)	%50 1.74 cm^2
Maliyet	0.52 $	1.00 $

5.4.3 Programlanabilir donanım birimlerinin In-System Programming (ISP) yöntemi ile programlanması

Programlanabilir donanım birimlerinin, devreye montajlarından sonra da programlanabilmeleri özelliği "In-System Programming" (ISP) olarak tanımlanmaktadır. ISP özelliğine sahip programlanabilir donanım birimlerinin kullanımı ile tasarım aşamasında giriş ve çıkış özelliklerinin belirlenmesinden sonra devrenin tasarımı ve üretimi paralel yürütülebilmektedir. ISP sayesinde tasarımda oluşabilecek hatalardan üretim etkilenmemekte ve oluşan hatalar, programlanabilir donanım biriminin devre üzerinde yeniden programlanması ile giderilmektedir. Tasarımın giriş ve çıkış özelliklerinin değişmemesi şartıyla kullanılan programlanabilir donanım biriminin kapasitesine bağlı olarak ISP yöntemi devreye yeni özelliklerin eklenebilmesine de imkan vermektedir (Şekil 5.11). Normal programlanabilir donanım birimleri ile ISP yöntemi ile programlanabilir donanım birimlerinin üretim akışı içerisinde bulundukları aşamaların karşılaştırması ve üretim akışında elde edilen hızlanma ve kolaylıklar Şekil 5.12'de görülmektedir.



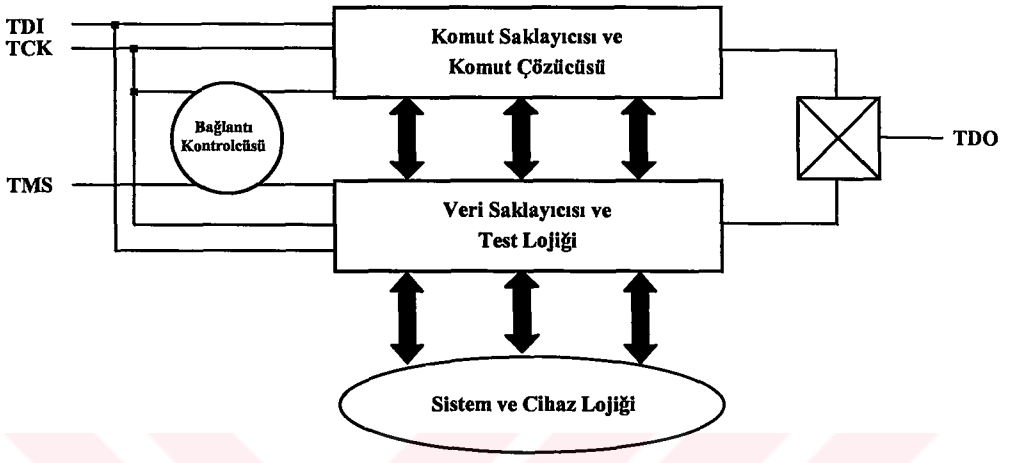
Şekil 5.11 ISP yöntemi ile tasarım ve üretimin beraber yürütülmesi (Altera)



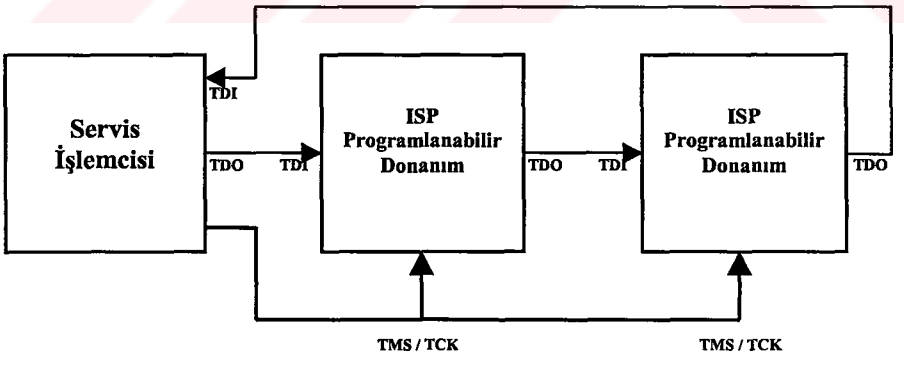
Şekil 5.12 Normal ve ISP programlanabilir donanım birimlerinin üretim akışı içinde bulundukları aşamalar (Altera)

Programlanabilir donanım birimleri, IEEE 1149.1 (JTAG, Joint Test Access Group – Birleşik Test Erişim Grubu) (Şekil 5.13) bağlantısı üzerinden ISP yöntemi ile programlanabilmektedirler. Dört adet sinyalden oluşan JTAG bağlantısı üzerinden programlanabilir donanım birimine, içerdiği fonksiyonel blokları programlamak amacı ile gerekli komut ve veriler aktarılabilmekte ve kontrol için birimin içeriği okunabilmektedir. JTAG bağlantısı üzerinden ISP gerçekleştirmek için tasarlanan devreye ya harici bir ISP bağlantısı eklenmesi ya da devrede JTAG protokolünü destekleyen özel bir servis işlemcisine yer verilmesi gerekmektedir. Gerçekleştirilen PCIP biriminde kullanılan programlanabilir donanım birimleri, ISP yöntemine uygun olarak programlanacak şekilde seçilmişler ve sistemin yapısında bulunan "Servis İşlemcisi" üzerinden ISP uygulanabilmesi sağlanmıştır. JTAG bağlantısı, daisy-chain (papatya-zinciri) yapısında

birden fazla birimin ardarda bağlanabilmesini desteklediği için sistemdeki tek servis işlemcisi üzerinden, gerçeklenen tasarımda kullanılan tüm programlanabilir donanım birimleri ISP yöntemi ile programlanabilmişlerdir. Servis işlemcisi üzerinden ISP uygulaması Şekil 5.14'de görülmektedir.



Şekil 5.13 IEEE 1149.1 JTAG bağlantısının prensip şeması



Şekil 5.14 Servis işlemcisi üzerinden ISP gerçekleştirilmesi

5.4.4 Programlanabilir donanımın çalışma sırasında yeniden yapılandırılması (Run-Time Reconfiguration)

Programlanabilir donanım birimlerinin içeriklerinin çalışma sırasında değiştirilebilmesi ve aynı programlanabilir donanımın farklı görevleri yerine getirmek amacı ile yeniden yapılandırılması "Run-Time Reconfiguration" olarak isimlendirilmektedir. Bu yöntemde programlanabilir donanım birimi, bir servis işlemcisinin kontrolünde gerektiğinde fonksiyonu değiştirilebilen bir yardımcı işlemci (co-processor) olarak kullanılmaktadır. İçerdikleri lojik ve flip-flop kaynaklarının sayısına ve yeniden yapılandırılma zamanlarına bağlı olarak programlanabilir donanım birimleri, run-time reconfiguration ile aynı devre üzerinde çok farklı görevleri yerine getirecek şekilde programlanabilmektedirler.

Bu tez çalışması kapsamında tasarlanan ve gerçekleştirilen PCIP biriminde değişik hızlar ve protokoller ile paralel kanala veri yollanmasını ve paralel kanaldan veri alınmasını sağlayan "Veri Akışı" altbiriminin gerçeklenmesinde kullanılan programlanabilir donanım birimi, run-time reconfiguration yöntemi ile ihtiyaç duyulduğunda karşılıklı kilitlemeli (basic interlocked), yüksek hızlı (high speed) ve veri akışı (data-streaming) protokolünü gerçekleştirecek şekilde çalışma sırasında yeniden yapılandırılmıştır.

5.4.4.1 ISP temelli run-time reconfiguration

Bu tez çalışmasında, run-time reconfiguration işlemini gerçekleştirmek için ISP yöntemini temel alan yeni bir yöntem geliştirilmiş ve bu yöntemin uygunluğu incelenmiştir. Geliştirilen yöntemin çalışması şu şekildedir:

1. Programlanabilir donanıma JTAG arabirimi üzerinden seri erişim

2. Servis işlemcisi ile programlanabilir donanımın 'test' ve 'program' moduna alınması

3. JEDEC formatındaki bilginin JTAG seri arabirimi üzerinden programlanabilir donanıma aktarılması

4. Programlama işlemi bittikten sonra programlanabilir donanımın normal çalışma moduna alınması

5. Yukarıdaki aşamaların çalışma sırasında gerektiğinde tekrarlanması

Bir run-time reconfiguration yönteminin uygunluğunun incelenmesinde yöntemin özel ek donanım veya yazılım gerektirmeden çalışabilmesi, standart yapılar kullanıyor olması ve bir yapılandırmadan diğer bir yapılandırmaya geçiş için ne kadar süre gerektirdiği önem taşımaktadır. Bu tez çalışmasında geliştirilen yeni yöntem, IEEE 1149.1 JTAG bağlantısı üzerinden programlanabilir donanıma eriştiği için hem uygulanabilirlik hem de değişik donanım platformlarına taşınabilirlik yönünden kolaylık sağlamaktadır. JTAG bağlantısının sadece dört sinyal üzerinden, özel ek donanım gerektirmeden gerçekleştirilebilmesi, geliştirilen yöntemin avantajlarındandır. Programlanabilir donanıma seri erişilmesi ve bu yüzden yeniden yapılandırma için paralel erişimli yöntemlere göre daha uzun zaman gerektiriyor olması uygulamanın türüne göre dezavantaj olarak görülebilir, fakat yeniden yapılandırma zamanının ön planda olmadığı uygulamalar için standartlara uygun olması, ek donanım gerektirmemesi ve kolay uygulanabilirliği geliştirilen yöntemin önemli özellikleridir. Kullanılan programlanabilir donanımın yapılandırma bilgisini elektrik kesintilerinde kaybolmayacak (EEPROM, FLASH) veya kaybolacak (SRAM) yöntemler ile saklıyor olması da geliştirilen yöntemin performansını ve uygunluğunu doğrudan etkileyen faktörlerdendir. Bu tez çalışmasında geliştirilen bu yöntem, gerçek zamanlı bir yapıya sahip olmasına rağmen, PCIP biriminin gerçekleştirilmesinde değişik paralel kanal protokolleri arasında problemsiz geçişi sağlayacak şekilde başarı ile uygulanmıştır.

5.4.4.2 ISP temelli run-time reconfiguration yönteminin yeniden yapılandırma zamanının değerlendirilmesi

Bu tez çalışmasında geliştirilen ISP temelli run-time reconfiguration yönteminin yeniden yapılandırma zamanının değerlendirilmesi için Vantis tarafından üretilen MACH serisi programlanabilir donanım ile Altera tarafından üretilen MAX serisi programlanabilir

donanımdan yararlanılmıştır. Buna göre her iki seri için geçerli olan yeniden yapılandırma zamanları Eşitlik (5.1) ve Eşitlik (5.2)'de verilmiştir.

$$t_p = t_e + \eta_r * t_{pw} + t_v \qquad (5.1)$$

t_p : yeniden yapılandırma zamanı

t_e : eski içeriğin silinme zamanı

η_r : programlanacak satır sayısı ($76 \le \eta_r \le 82$)

t_{pw} : programlama sinyali süresi (50 ms min.)

t_v : yeni içeriğin kontrol edilme süresi (300 ms max.)

$$t_p = t_{pulse} + \frac{Cycle_{PTCK}}{f_{TCK}} \qquad (5.2)$$

t_p : yeniden yapılandırma zamanı

t_{ppulse} : silme, programlama ve kontrol sırasında harcanan standart sabit zaman

$Cycle_{PTCK}$: programlama için gerekli çevrim sayısı

f_{TCK} : programlama frekansı

Vantis MACH serisi 82 satırı olan bir programlanabilir donanım biriminin bu tez kapsamında geliştirilen yöntem ile yeniden yapılandırılması yaklaşık 4.5 s sürmektedir. Aynı yöntem ile Altera MAX 9000, EPM9560 biriminin, 10 MHz programlama frekansı ile programlanması yaklaşık 12.15 saniye tutmaktadır.

6. YAZILIM ÖZELLİKLERİ

Bu tez çalışmasında, ana bilgisayar sistemlerindeki paralel kanal yapısının PCI yerel yoluna bağlanması için önerilen sistemin tasarım ve gerçekleştirilmesinde donanımın yanısıra yazılımlardan da yararlanılmıştır. Bu bölümde, gerçekleştirilen yazılım modülleri incelenecek ve bu modüllerin ayrıntılarına yer verilecektir.

6.1 Servis İşlemcisi Kontrol Programı

Protokol katmanına ve dönüşüm katmanının servis işlemcisinde gerçeklenmiş bölümüne ait yazılım modüllerinin çalıştırılması, bu modüller arasında senkronizasyonun sağlanması, donanım ve yazılım kaynaklarına erişimin kontrol edilmesi ve bu kaynakların paylaştırılması amacı ile servis işlemcisinde çalıştırılmak üzere, gerçek-zamanlı çalışmaya uygun, "Service Processor Control Program" (**SPCP**) adı verilen özel bir kontrol programı gerçekleştirilmiştir.

Kontrol programı, servis işlemcisinin donanım ve programlama modeline uyum sağlayacak ve çoklu programlama (multiprogramming) ortamını destekleyecek yapıda tasarlanmış ve kontrol programı tarafından çalıştırılabilecek en küçük program parçası olarak proses yapısı seçilmiştir. SPCP kontrolünde çalışan bir proses için bulunabileceği 3 durum tanımlanmıştır (Şekil 6.1):

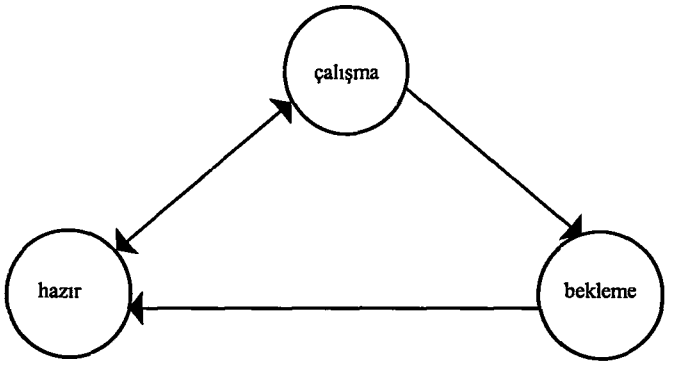
1. hazır durumu

2. çalışma durumu

3. bekleme durumu

Çalışmaları için gerek tüm kaynaklara sahip olan prosesler, hazır durumunda işlemciyi kullanma hakkının kendilerine gelmesini beklemektedirler. İşlemci tarafından işletilen proses çalışma durumundadır. Çalışmaya devam edebilmek için erişmek istedikleri

kaynağın serbest kalmasını veya başlattıkları bir işlemin sonuçlanmasını bekleyen prosesler ise bekleme durumundadırlar.



Şekil 6.1 SPCP'de bir prosesin bulunabileceği 3 durum

Kontrol programının tasarımında, bu tez çalışmasında gerçekleştirilen sistemin çalıştırılmasına yönelik olması göz önüne alınarak hem kaynakları daha iyi kullanmak hem de performansı arttırmak amacıyla aşağıdaki kriterler belirlenmiş ve uygulanmıştır.

6.1.1 Kontrol programının desteklediği proses sayısı

Kontrol programının desteklediği proses sayısı 15 ile sınırlıdır. Servis işlemcisi olarak kullanılan Intel 80960CA işlemcisi, aktif saklayıcı kümesinin bir prosesten diğerine geçilmesi anında kaydedilmesi amacı ile donanım seviyesinde işlemci içerisinde bulunan 15 arabelleğin kullanılmasına izin vermektedir. 16. arabelleğe ihtiyaç duyulması durumunda aktif saklayıcı kümesinin işlemci içerisinde saklanabileceği arabellek kalmamakta ve çalışmakta olan programın aktif saklayıcı kümesini dış belleğe kaydetmesi gerekmektedir. Ayrıca bu durumda işlemci bir prosesten diğerine saydam bir şekilde donanım olarak geçememektedir. Hem sistem performansını düşürmemek hem de dış belleğe saklayıcıların kaydedilmesinde gereken bellek alanını belirlemek için dinamik bellek kullanımının oluşturacağı yükü ortadan kaldırmak amacıyla 15 proses sınırının aşılmamasına karar verilmiştir.

6.1.2 Kontrol programında proseslerin oluşturulması

Servis işlemcisinde çalışan kontrol programı, bu tez çalışmasında gerçekleştirilen sisteme özel tasarlandığı için, geliştirme aşamasında, servis verdiği katmanlar için hangi yazılım modüllerine ihtiyaç duyulduğu belirlenmiş ve bu modüllerden oluşan proses yapısı çalışma sırasında değişmeyecek şekilde oluşturulmuştur. Kontrol programının çalışması sırasında yeni proseslerin oluşturulması öngörülmemiştir. SPCP, açılış anında tanımlı olan tüm prosesleri çalışmaya hazır duruma getirip, bekleme konumuna alacak şekilde tasarlanmıştır. Bu özellik sayesinde çalışma sırasında bir prosese ihtiyaç duyulduğunda, proses oluşturma zamanı kaybedilmemekte ve kontrol programının cevap süresi iyileşmektedir.

6.1.3 Kontrol programında proses yetki seviyeleri

Servis işlemcisinin programlama modelinde, proseslerin işlemci içindeki kaynaklara ve iç veya dış bellek bölgelerine erişimlerini derecelendirmek ve gerektiğinde engellemek amacı ile "kullanıcı" ve "sistem" yetki seviyeleri tanımlanmış olmasına rağmen, geliştirilen kontrol programında, kontrol karmaşıklığını en aza indirmek ve "kullanıcı" - "sistem" yetki seviyeleri arasında gereksiz geçişleri ve bu geçişlerden kaynaklanacak gecikmeleri engellemek için, tanımlı olan tüm prosesler kaynaklara erişim ve kullanım yönünden eşdeğer kabul edilmişlerdir. Kontrol progamında, tanımlı olan tüm prosesler "sistem" yetki seviyesinde çalışacak ve ortak bir adres alanı içerisinde farklı alanları kullanacak şekilde gerçeklenmişlerdir.

Çalışma sırasında yeni proseslerin oluşturulmaması ve SPCP kontrolünde çalışan proseslerin, kullanacakları kaynaklar göz önünde bulundurularak birbirlerinin çalışmalarını bozmayacak şekilde geliştirilmiş olmaları sayesinde, tanımlı tüm proseslerin "sistem" yetki seviyesinde çalıştırılmaları, SPCP içinde problemlere sebep olmamaktadır.

6.1.4 Kontrol programında proseslerin çalıştırılması

SPCP, istek-cevap modeline uygun olarak paralel kanaldan veya uygulama katmanından gelen istekleri cevaplamak için ilgili prosesleri çalıştıracak şekilde düzenlenmiştir.

İstek-cevap yapısında proseslerin çalıştırıldığı uygulamalarda, işletim sisteminin, çalışmakta olan prosesi, belirli bir sürenin (time-quantum) sonunda, durdurması ve çalışmaya hazır diğer bir prosesi çalışır konuma alması şeklinde prosesler arasında geçişi sağlaması (pre-emptive scheduling) yerine proseslerde çalışma anında oluşabilecek hatalara karşı sistemi korumak amacı ile maksimum bir süre aşılmadığı sürece, proseslerin işlerini tamamlayıp kendi istekleri ile işlemciyi bir başka prosese devretmeleri yöntemi (non pre-emptive scheduling) ile prosesler arası geçişi sağlaması sistemin performansını arttırmaktadır.

Geçekleştirilen kontrol programında, sistemin istek-cevap modelinde çalışması göz önüne alınarak, prosesler arasında geçişi sağlamak için ikinci yöntemin uygulanmasına karar verilmiştir. Bu yöntem, "ortak çoklu programlama" (cooperative multiprogramming) olarak da adlandırılmaktadır.

SPCP kontrolünde çalışan proseslerin, işlemci kontrolünü izin verilenden daha uzun süre elinde bulundurmasını ve sistemin çalışmasını olumsuz etkilemesini engellemek amacı ile gerçeklenen kontrol yöntemi izin verilen sürenin aşılması durumunda kesme oluşturulmasını ve kontrolün SPCP'ye aktarılması sağlanmaktadır.

6.1.5 Kontrol programında kesme isteklerinin değerlendirilmesi

Kontrol programının, paralel kanaldan veya dönüşüm katmanının merkezi işlemcide gerçeklenen bölümünden gelen istekleri farketmesi için kesme sinyallerinden yararlanılmıştır. Bu sayede, "Ortak çoklu programlama" yapısına uygun bir şekilde, öncelikli servis verilmesi gererken bir durum oluştuğunda aktif prosesin çalışmasının gelen kesme isteği ile durdurulması, ilgili kesme servis programının devreye girmesi sağlanmıştır.

Proseslerin çalışmasını bölme hakkına sahip özel durumlar belirlenmiş ve bu durumlara servis verecek kesme servis prosesleri oluşturulmuştur. Kesme servis proseslerine de aralarında, verdikleri servisin bir başka kesme servis prosesi tarafından bölünüp bölünemeyeceğine göre öncelikler verilmiştir. Bu öncelik yapısının uygulanabilmesi için, servis işlemcisinin donanım olarak içerdiği kesme önceliklerinden yararlanılmıştır. Servis işlemcisi kesme istekleri arasında öncelik yapısı oluşturduğu için, bir prosesin çalışması sırasında geçerli olan kesme isteği seviyesinin altındaki seviyelerde oluşan kesmeler prosesi etkilememekte, daha yüksek seviyeli kesmeler de ise işlemci aktif prosesi durdurup, kesme servis programına dallanılmasını sağlamaktadır. Bu özelliği kullanmak amacıyla, bir prosese ait kesme seviyesi değeri, o proses çalıştırılacağı zaman, işlemcinin özel saklayıcılarına yerleştirilmektedir. Bu sayede donanım olarak düşük öncelikli kesme isteklerin çalışan proses yansıtılmaması, yüksek öncelikli kesme isteklerinin ise servis görebilmesi sağlanmaktadır.

Paralel kanaldan gelen isteklerin belirlenebilmesi için Operational Out ve Address Out sinyallerinin oluşturduğu kesmelerden, dönüşüm katmanının merkezi işlemcide çalışan bölümünden gelen isteklerin anlaşılıp servis verilebilmesi için protokol katmanında yer alan "doorbell" kesmesinden yararlanılmıştır. Gerçekleştirilen kesme yapısı, paralel kanaldan gelen isteklere öncelik verecek şekilde düzenlenmiş ve öncelik sırasıyla Operational Out kesmesi, Address Out kesmesi ve "doorbell" kesmesi değerlendirilmiştir.

Operational Out sinyali, paralel kanalın aktif olup olmadığını gösterdiği ve pasif olması durumunda kanala bağlı tüm kontrol birimlerinin 1.5 μs içerisinde sinyallerini geri çekmelerini gerektirdiği için en öncelikli kesme isteği olarak tanımlanmıştır.

Address Out sinyali, 5. bölümde anlatıldığı gibi, 1 μs içerisinde, kanaldan gelen adresin kayıtlı adresler içerisinde bulunup bulunmadığının belirlenmesi ve kayıtlı olmaması durumunda Select Out sinyalinin bir sonraki kontrol birimine aktarılmasının sağlanabilmesi için ikinci öncelikli kesme isteği olarak değerlendirilmiştir.

"Doorbell" kesmesi ise üçüncü öncelikli kesme isteği olarak, merkezi işlemciden gelen isteklerin farkedilmesi için kullanılmıştır.

6.2 Protokol Katmanını Oluşturan Yazılım Modülleri

Protokol katmanının, hem paralel kanal hem de PCI yerel yoluna erişebilmesi için gerçekleştirilen donanım birimlerini kontrol etmek üzere SPCP altında çalışan modüller oluşturulmuştur.

PCI yerel yoluna erişim için kullanılan PCI 9060 biriminin PCI protokolünü donanım olarak desteklemesi, yazılımın müdahelesine gerek olmadan yerel yola erişim hakkını elde etmesi, veri aktarımı için gerekli sinyalizasyonu yapması sayesinde, protokol katmanında PCI yerel yolu ile ilgili sadece verileri bir adres bölgesinden diğerine aktaran yazılım modüllerinin oluşturulması yeterli olmuştur.

Paralel kanal ile bağlantıyı sağlamak, adresleme ve veri aktarım işlemlerini yürütmek amacı ile bu tez çalışması kapsamında gerçekleştirilen "Parallel Channel Interface Processor", birimi yazılımla kontrol edildiği için, protokol katmanında paralel kanal protokolünü uygulamak amacıyla değişik modüller geliştirilmiştir.

6.2.1 Protokol katmanında paralel kanal ile ilgili yazılım modülleri

PowerUp modülü, gerçekleştirilen sistemin, Select Out sinyalinin elektriksel sürekliliğini kesmeden paralel kanala bağlanabilmesini ve kanal ile fiziksel bağlantı kurabilmesini sağlar. PowerUp modülü, Şekil 5.6'da görülen K1, S1 ve S2 rölelerini kontrol ederek kanal ile bağlantıyı kurar. Bunun için:

1. S2 rölesini çekerek, çıkış sürücülerinin devreye girmesini sağlar.

2. S1 rölesini çekerek, girişteki Select Out sinyalinin PCIP birimi ile bağlantısını sağlar.

3. K1 rölesini çekerek, Select Out sinyalinin çıkışa röle üzerinden aktarılmasını sonladırır.

PowerDown modülü, gerçekleştirilen sistemin, Select Out sinyalinin elektriksel sürekliliğini kesmeden paralel kanal ile fiziksel bağlantının sonladırılmasını sağlar. Bu amaçla Şekil 5.6'da görülen K1, S1 ve S2 rölelerini kontrol ederek

1. Yürümekte olan mantıksal kanal işlemlerinin sonladırılması gerektiğini ilgili proseslere bildirir

2. K1 rölesini bıraktırarak, Select Out sinyalinin girişten çıkışa mekanik olarak aktarılmasını sağlar

3. S1 rölesini bıraktırarak, Select Out sinyalinin PCIP birimi ile bağlantısını sonlandırır

4. S2 rölesini bıraktırarak, çıkış sürücülerinin devre dışı kalmasını sağlar

GenerateChannelSequence modülü, paralel kanal ile bağlantı sırasında paralel kanal sinyallerinin geçerli kombinasyonlarının kullanılması ile özel durumların belirtilmesini gerçekleştirir. Örneğin PCIP birimi bağlı olduğu kanalın bir seçme işlemini başlatmasını sağlamak amacıyla Request In sinyalinin aktif hale getirilmesini, Select Out sinyali aktif iken yürümekte olan bir seçme işleminde kanaldan gelen isteği geçici meşguliyetinden

dolayı kabul edemeyeceğini belirten Status In sinyalinin aktif hale getirilmesini ve benzer durumları bu modül ile gerçekleştirir.

AcceptChannelSequence modülü, paralel kanaldan gelen sinyal kombinasyonlarının anlamını çözmek ve paralel kanal protokolüne uygun cevabı oluşturmak için geliştirilmiştir. Paralel kanaldan gelen isteklerin zamanında farkedilebilmesi ve tanımlı süreler içinde cevap verilebilmesi amacıyla bu modül kendi içerisinde alt modüllerden oluşturulmuştur. Her bir alt modül de kontrol edilen paralel kanal sinyalinin bağlı olduğu kesme girişine servis veren kesme servis prosesi olacak şekilde gerçekleştirilmiştir.

ToChannel modülü, PCIP biriminden paralel kanala veri aktarımını sağlamak amacı ile gerçekleştirilmiştir. Bu modül, parametre olarak hangi veri aktarım yönteminin kullanılacağını ve hangi arabellek adresinden ne uzunlukta veri aktarılacağını kabul edecek şekilde oluşturulmuştur. Veri akışı yöntemi dışındaki aktarımlarda program kontrollü aktarım yapılması, veri akışında ise gerçekleştirilen veri akışı donanımının programlanması sağlanmıştır. ToChannel modülü veri akışını başlatmadan önce, bu amaç için oluşturulmuş arabelleği dolduracak kadar veriyi arabelleğe aktarır ve veri akışı donanımını aktif hale getirir. Veri akışı donanımı, paralel kanala bilgileri arabellekten alarak yollar. Arabellek yarısına kadar boşaldığında, oluşturduğu kesme sinyali ile ToChannel modülünün, arebellekte boşalan bölgeyi doldurmasını sağlar. Bu özellik sayesinde ToChannel modülünün, veri akışını başlattıktan sonra, kontrolü servis işlemcisine bırakması ve çalışmayı bekleyen diğer proseslere zaman tanıması sağlanmıştır.

Veri aktarımında 4.5 Mbyte/s hızı ile çalışılırken arabellekten yeni veri alınıp paralel kanala aktarılma periyodu 222 ns dir. Bu durumda gerçekleştirilen sistemde 64 KB'lık arabellek kullanıldığı göz önüne alınırsa, veri akışı alt biriminin 32 KB uzunluğunda veriyi kanala aktarması 7281 μs sürecektir. Servis işlemcisi 33 MHz sistem saatinde, bir çevrimde ortalama 2 komut işleyecek yapıda olduğu için, bu zaman diliminde diğer proseslere 485400 komutluk bir süre kalmasını sağlamaktadır. Bu sayede yüksek hızlı veri aktarımının, sistem performansını etkilemeden, gerçek-zamanlı çalışmaya uygun şekilde yapılması sağlanmıştır.

FromChannel modülü, paralel kanaldan PCIP birimine veri aktarımını gerçekleştirmek üzere oluşturulmuştur. Bu modülün çalışma prensibi, ToChannel modülünün çalışma prensibi ile aynıdır. FromChannel modülü de veri akışı yönteminde, gerçekleştirilen donanımdan yararlanacak şekilde tasarlanmıştır. ToChannel çalışması sırasında uygulanan zamanlamanın aynısı FromChannel çalışmasına da uygulanmıştır. Paralel kanaldan gelen veriler, arabelleğin yarısını doldurduklarında, arabellek oluşturduğu kesme isteği ile FromChannel prosesinin bu durumun farkına varmasını sağlamaktadır.

CommandDecode modülü, paralel kanaldan gelen giriş_çıkış komutlarını çözmek, geçerliliklerini belirlemek ve bu komutları dönüşüm katmanına aktarılacak duruma getirmek için yazılmıştır. Bu modül bir komuta ait kaç parametrenin olduğunu da belirleyerek bu parametrelerin paralel kanaldan okunmasını sağlar.

6.2.2 Protokol katmanında PCI yerel yolu ile ilgili yazılım modülleri

ToPCI modülü, kaynağı, hedefi ve uzunluğu belirtilen bir bilginin PCI yerel yolundaki bir başka birime aktarılması için gerçeklenmiştir. Temel olarak bellekte bir adres bölgesinden bir başka adres bölgesine kopyalama işlemi olarak çalışır.

FromPCI modülü, kaynağı, hedefi ve uzunluğu belirtilen bir bilginin PCI yerel yolundaki bir birimden PCIP'ye aktarılması için gerçeklenmiştir. Çalışma prensibi ToPCI ile aynıdır.

6.3 Dönüşüm Katmanını Oluşturan Modüller

Dönüşüm katmanının hem servis işlemcisinde hem de merkezi işlemcide gerçekleştirilmiş olması sebebiyle bu katmana ait modüllerin bir bölümü SPCP'nin diğer bölümü ise kişisel bilgisayardaki işletim sisteminin kontrolünde çalışmaktadır.

NotifyUXL modülü, SPCP altında gerçeklenmiştir ve dönüşüm katmanının servis işlemcisinde çalışan bölümünün, merkezi işlemcide çalışan bölüme erişmesinde kullanılır. Bu modül, protokol katmanında gerçekleştirilen "doorbell" saklayıcılarını kullanarak

dönüşüm katmanının üst kısmına kesme yolanmasını ve "mailbox" saklayıcılarında da parametrelerin aktarılmasını sağlar.

NotifyLXL modülü, merkezi işlemcide gerçeklenmiştir ve dönüşüm katmanının servis işlemcisinde gerçeklenen bölümüne istek aktarılmasında kullanılır. NotifyLXL modülü de, "doorbell" ve "mailbox" saklayıcılarından yararlanır.

RespondUXL modülü, NotifyLXL modülünden gelen istekleri değerlendirmek amacı ile oluşturulmuştur. "Doorbell" saklayıcının oluşturduğu kesme isteğini değerlendirerek, dönüşüm katmanının üst kısmından bir istek geldiğini belirler.

RespondLXL modülü, merkezi işlemcide çalışmaktadır ve dönüşüm katmanının alt kısmında NotifyUXL ile gönderilen istekleri değerlendirmek amacı ile gerçeklenmiştir. Bu modül, "doorbell" saklayıcısının merkezi işlemci tarafında oluşturduğu kesme istekleri ile dönüşüm katmanının alt kısmından istek geldiğini belirleyecek şekilde geliştirimiştir.

XlateDevice modülü, SPCP altında gerçeklenmiştir ve ana bilgisayar çevre birimleri ile kişisel bilgisayar çevre birimleri arasındaki geçişi sağlar. Uygulama katmanında sonuçlanan giriş_çıkış işlemlerine bağlı olarak, ilgili çevre biriminde gerekli alanların güncellenmesini ve bu alanların yapısının ana bilgisayar çevre birimlerinde tariflenenlerle uyumlu ve tutarlı olmasını sağlar.

7. GERÇEKLEŞTİRİLEN SİSTEMİN PERFORMANSININ DEĞERLENDİRİLMESİ

Bu tez çalışması kapsamında tasarlanan ve gerçekleştirilen sistemin performansının değerlendirilebilmesi için PCI yerel yolu ile paralel kanal performansının incelenmesi ve değerlendirilmesi gerekmektedir.

Tasarlanan ve gerçekleştirilen sistem, ana bilgisayar sistemleri ile kişisel bilgisayar sistemlerinin entegrasyonunu fiziksel seviyede gerçekleştirdiği, paralel kanal ile PCI yerel yolu arasında donanım aracılığı ile bağlantı kurduğu ve istemci_sunucu modelinden farklı olarak ana bilgisayar sisteminin, kişisel bilgisayar çevre birimlerini arada üst seviyeli haberleşme protokolleri (TCP/IP, NFS, vb.) olmadan kullanabilmesini sağladığı için hem paralel kanal hem de PCI yerel yolunda geçerli olan performans kriterleri, gerçekleştirilen sistemin performansını doğrudan etkilemektedir.

Gerçekleştirilen sistemin donanımının tasarımında ve gerçeklenmesinde, PCI yerel yolundan paralel kanala ve paralel kanaldan PCI yerel yoluna geçişlerin bekleme çevrimine gerek olmadan yapılabilmesi için programlanabilir donanım birimlerinden yararlanılmış, bu birimlerin sağladığı yüksek yoğunluk sayesinde donanım fonksiyonları mümkün olduğunca aynı donanım birimleri içerisinde gerçeklenmiş ve bu sayede oluşabilecek yayılım gecikmeleri en aza indirilmiştir. Programlanabilir donanım kullanımı, yazılım fonksiyonlarının da uygun olanlarının donanıma aktarılmasını ve donanım ile yürütülerek servis işlemcisinin diğer işlemlere zaman ayırabilmesini ve gerçekleştirilen sistemin donanım ve yazılım seviyesinde maksimum paralellik ile çalıştırılmasını sağlamıştır.

PCI yerel yolunun 33 MHz sistem frekansında çalıştığı ve 32 bitlik veri yolu üzerinden 132 Mbyte/s hızında veri aktarımı yaptığı değerlendirilecek olursa paralel kanaldan 4.5 Mbyte/s hızı ile gelen isteklerin değerlendirilmesi için 8 bitlik veri aktarımı temel alındığında 1:7 oranında bir işlem zamanı avantajı olduğu görülmektedir. Bu durumda tasarlanan ve gerçekleştirilen sistem, paralel kanaldan gelen isteklerin yoğunluğu ile bir

darboğaza girmeyecektir. PCI yerel yolundan paralel kanala veri aktarımında da PCI yerel yolu tarafında uygulanan akış kontrolü (flow control) ve arabellek kullanımı ile PCI yerel yolunda darboğaz oluşması engellenmiştir.

7.1 PCI Yerel Yolu Performansının Değerlendirilmesi

PCI yerel yolunun performansının değerlendirilmesinde ve ifade edilmesinde, farklı sistemler ve tasarımlar arasında anlamlı ve tutarlı karşılaştırmalar yapılmasını sağlayan kriterler tanımlanmıştır. Bu kriterler ve tanımlamaları Tablo 7.1'de görülmektedir.

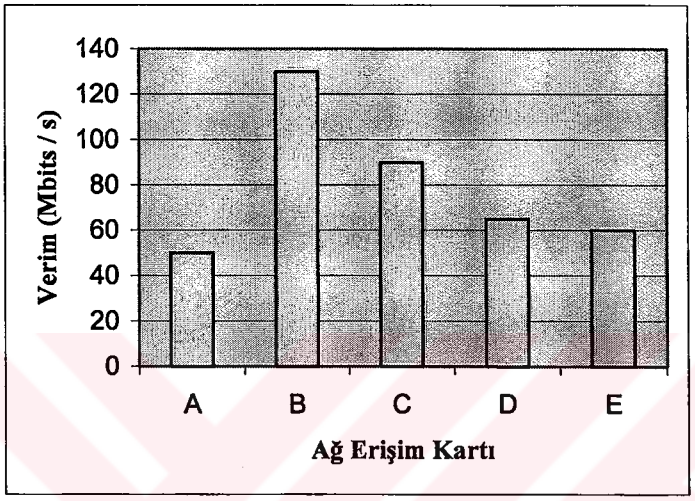
Tablo 7.1 PCI yerel yolu performasının değerlendirilmesinde kullanılan kriterler ve tanımları

Kriter	Tanımı
PCI kullanımı (PCI utilization)	PCI yerel yoluna erişimde aktif olarak kullanılan çevrim sayısının toplam PCI çevrim sayısına oranı. FRAME#, IRDY# veya TRDY# sinyallerinden birinin aktif olması, PCI aktif çevrimini göstermektedir.
PCI verimi (PCI throughput)	Birim zamanda, PCI yerel yolu üzerinden aktarılan veri miktarı. Aktarılan veri, uygulamaya ait veri ve PCI protokolüne ait kontrol verisi olamk üzere iki kısımdan oluşmaktadır. Kontrol verisi, uygulama verisi yanında ihmal edilebilecek orandadır.
PCI efektif kullanımı (PCI efficiency)	Bir PCI biriminin, PCI kullanımını minimize ederken PCI verimini ne kadar maksimize ettiğini gösterir. PCI veriminin, PCI kullanımına oranı olarak ifade edilir.

PCI efektif kulllanımı, diğer iki kriterden farklı olarak PCI yerel yolunun kullanım oranından bağımsız bir değer belirtmektedir. Bu özelliği ile PCI efektif kullanımı, birbirinden bağımsız ve farklı PCI tasarımlarının performanslarının karşılaştırılabilmesini sağlamakta ve sistemlerin PCI performansının ne ölçüde ölçeklenebilir olduğunu göstermektedir. PCI efektif kullanımı, "1" değerine yaklaştıkça performansın arttığını, "0" değerine yaklaştıkça, ana birim veya hedef birim bekleme çevrimleri, kontrol çevrimleri

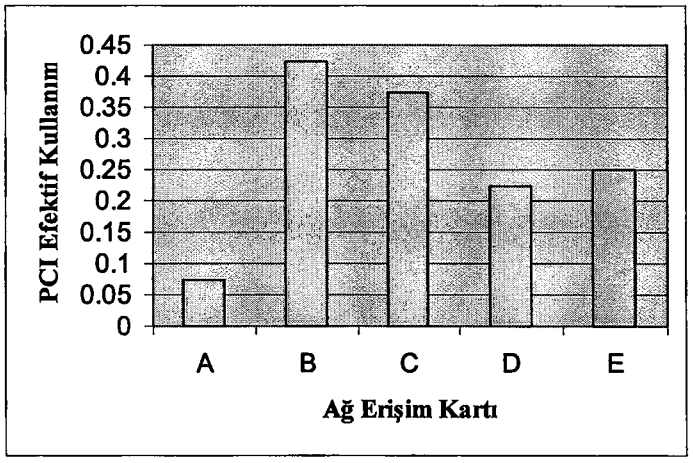
veya yeterince arabelleklenmemiş veri aktarımları yüzünden performansın düştüğünü belirtmektedir.

PCI performansını belirtilen kriterlere göre değerlendirmek amacı ile 5 farklı "Ağ Erişim Kartı" (NIC) 'nın PCI yerel yolundaki çalışmalarının incelenmesi sonucunda kartlar için elde edilen PCI verim değerleri Şekil 7.1'de görülmektedir.



Şekil 7.1 5 farklı "Ağ Erişim Kartı"'na ait PCI verim değerleri

Aynı kartlar için geçerli olan PCI efektif kullanım değerleri ise Şekil 7.2'de verilmiştir.

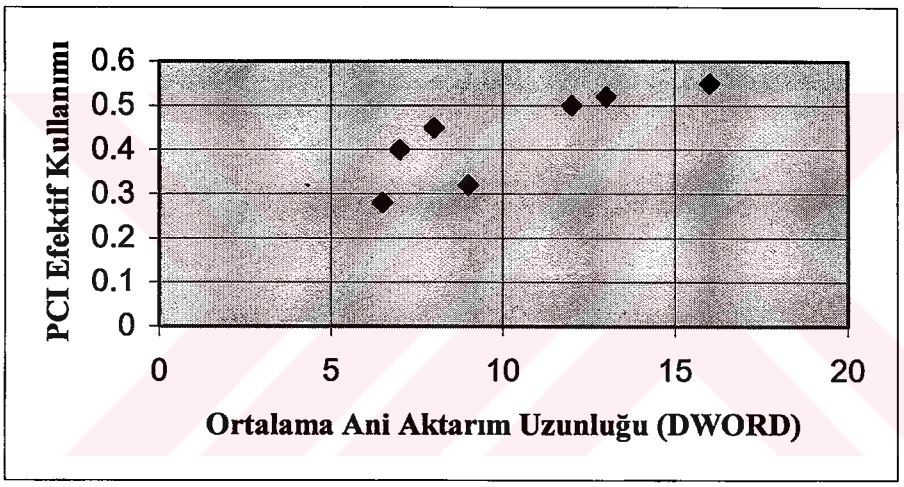


Şekil 7.2 5 farklı "Ağ Erişim Kartı"'na ait PCI efektif kullanım değerleri

PCI birimlerinin ani veri aktarımı (burst data transfer) sırasında aktardıkları veri miktarı da PCI performansını etkilyen önemli faktörlerdendir. Her PCI birimi, X adet veriyi aktarmak için PCI yerel yoluna eriştiğinde aktarım işlemi başlayıncaya kadar L çevrim beklemek zorunda olduğuna göre, her PCI çevriminde bir veri aktarıldığı varsayılırsa, aktarımın efektifliği

$$\frac{X}{X+L}$$

olacaktır. L parametresi, sistem tarafından belirlendiğinden ve PCI birimin tarafından değiştirilemediğinden, efektif kullanım değerini arttırmak için PCI biriminin X değerini maksimuma çekmesi gerekmektedir. Şekil x.x'de PCI efektif kullanımı ile bir seferde aktarılan veri uzunluğu arasındaki ilişki görülmektedir.



Şekil 7.3 Ortalama ani aktarım uzunluğu ile PCI efektif kullanımı arasındaki ilişki

7.2 Paralel Kanal Performansının Değerlendirilmesi

Paralel kanal performansının değerlendirilmesinde etkili olan kriterlerin sayısı PCI yerel yolundaki kriterlere göre daha fazladır. Paralel kanal, merkezi işlemcinin kontrolündeki kanal işlemcileri ile çevre birimleri arasında veri aktarımını sağladığı ve PCI yerel yolu ile karşılaştırıldığında fiziksel protokolün üstünde yazılım destekli ikinci bir seviye iletişim

Tablo 7.2 Paralel kanal performansını belirleyen kriterler ve tanımları

Kriter	Tanım
SSCH+RSCH oranı (SSCH+RSCH rate)	Çevre biriminde, saniyede başlatılan giriş_çıkış işlemi sayısı
Meşgul olma yüzdesi (Pct Busy)	Belirli bir zaman diliminde, erişilen çevre biriminin meşgul olma oranı
Bekletme zamanı (Time Pend)	Kanal işlemcisine giriş_çıkış komutunun yollanması ile bu komutun ilgili çevre biriminde başlatılması arasında geçen süre
Bağlantının kesik kalma süresi (Time Disc)	Fiziksel hareket gerektiren (diskte arama vb.) veya yürütülmesi belli bir zamanı aşan giriş_çıkış işlemlerinde kanal tarafından çevre biriminde işlem başlatılmasından sonra, çevre biriminin kanala tekrar bağlanmasına kadar geçen ortalama süre
Bağlı kalma süresi (Time Conn)	Kanal ile çevre birimi arasında veri aktarımı için geçen ortalama süre
Servis süresi (Time Serv)	Çevre biriminin istenen giriş_çıkış işlemini gerçekleştirme süresi
Cevap süresi (Time Resp)	Çevre biriminin gerçekleştirdiği giriş_çıkış işlemine cevap verme süresi
Giriş_Çıkış kuyruğu ortalama uzunluğu (SSCHs in Queue, Mean)	Çevre birimlerinde yürütülmek üzere kuyrukta bekleyen ortalama giriş_çıkış isteği
Giriş_Çıkış kuyruğu maksimum uzunluğu (SSCHs in Queue, Max)	Belirli bir zamanda giriş_çıkış kuyruğunda bekleyen maksimum giriş_çıkış isteği sayısı

Paralel kanal için belirtilen kriterlere göre paralel kanalın disk biriminden teyp birimine veri aktarımındaki performans değerleri, Tablo 7.3 ve Tablo 7.4'de, disk birimleri arasında veri aktarımı sırasındaki performans değerleri ise Tablo 7.5 ve Tablo 7.6'da görülmektedir.

Tablo 7.3 Disk birimi ile teyp birimi arasında giriş_çıkış işlemi sırasında kanallar için bağıl meşgul olma dağılımı

```
PRF013  Run 12/16/1999 21:44:12          CHANNEL_BUSY                                    Page    3
                                          Channel Busy Relative Distribution
 From 12/16/1999 21:35:13                                                                TRYILDIZ
 To   12/16/1999 21:43:13                                                  CPU 9121       SN  30732
 For    480 Secs 00:07:59                 YTU CSE DEPT. VM/ESA V2.2        ES/VM     20.02 SLU 0000
```

Chan in Dec	Chan in Hex	Pct Chan Busy	Channel Busy Percent									
			Busy =0	0< Busy <=10	10< Busy <=20	20< Busy <=30	30< Busy <=40	40< Busy <=50	50< Busy <=60	60< Busy <=70	70< Busy <=80	80< Busy <=90	90< Busy <=100
13	(0D)	64.1	13	13	0	0	0	0	0	0	25	50	0
5	(05)	6.1	0	88	13	0	0	0	0	0	0	0	0
8	(08)	6.1	25	38	38	0	0	0	0	0	0	0	0
4	(04)	0.6	75	25	0	0	0	0	0	0	0	0	0
9	(09)	0.2	88	13	0	0	0	0	0	0	0	0	0

Tablo 7.4 Disk birimi ile teyp birimi arasında giriş_çıkış işlemi sırasında disk aktivitesinin dağılımı

```
PRF012  Run 12/16/1999 21:44:11          DASD_BY_ACTIVITY                                    Page    2
                                          DASD Activity Ordered by Activity
 From 12/16/1999 21:35:13                                                                   TRYILDIZ
 To   12/16/1999 21:43:13                                                  CPU 9121       SN  30732
 For    480 Secs 00:08:00                  YTU CSE DEPT. VM/ESA V2.2       ES/VM     20.02 SLU 0000
```

<-----Device------>					<--SSCH+RSCH-->			<--------------Time-------------->					<-SSCHs in Queue-->		
Number	Volume Serial	Type	Control Unit	Owner	Mini-disk Links	On-line Secs	Count	Rate	Pct Busy	Pend	Disc	Conn	Serv	Resp	Mean	Max	Err
0102	ESAW02	3390-2	3990*EC		153	480	1360	2.8	5.4	0.3	1.8	16.9	19.0	19.0	0	0	0
0100	ESAW00	3390-2	3990*EC		147	480	1168	2.4	4.6	0.3	2.2	16.3	18.8	19.7	0.0	0.0	0
0101	ESAW01	3390-2	3990*EC		184	480	1064	2.2	4.5	0.3	3.4	16.7	20.4	20.4	0	0	0
0103	ESAW03	3390-2	3990*EC		79	480	181	0.4	0.5	0.3	2.8	10.5	13.6	13.6	0	0	0
0117	ESAW07	3390-1	3990*EC		7	480	51	0.1	0.2	0.3	1.4	12.7	14.4	14.4	0	0	0
0115	ESAW05	3390-1	3990*EC		6	480	48	0.1	0.1	0.3	1.6	12.3	14.2	14.2	0	0	0
0114	ESAW04	3390-1	3990*EC		1	480	23	0.0	0.0	0.3	2.1	1.1	3.4	3.4	0	0	0
0116	ESAW06	3390-1	3990*EC		2	480	16	0.0	0.0	0.3	0.1	0.8	1.1	1.1	0	0	0
0118	ESAW08	3390-1	3990*EC		2	480	16	0.0	0.0	0.3	0.0	0.8	1.1	1.1	0	0	0
0119	XXX059	3390-1	3990*EC		0	480	16	0.0	0.0	0.3	0.0	0.8	1.1	1.1	0	0	0
011A	XXX05A	3390-1	3990*EC		0	480	16	0.0	0.0	0.3	0.0	0.8	1.1	1.1	0	0	0
011B	XXX05B	3390-1	3990*EC		0	480	16	0.0	0.0	0.3	0.0	0.8	1.1	1.1	0	0	0
The table above contains the top					12 of	12 items.											
Sum/Mean					48	5760	3975	0.7	1.3	0.3	2.3	15.9	18.5	18.8	0.0	0.0	0

Tablo 7.5 Disk birimleri arasında giriş_çıkış işlemi sırasında kanallar için bağıl meşgul olma dağılımı

```
PRF013  Run 12/16/1999 22:00:50          CHANNEL_BUSY                                    Page    3
                                          Channel Busy Relative Distribution
 From 12/16/1999 21:50:31                                                                TRYILDIZ
 To   12/16/1999 21:59:31                                                  CPU 9121      SN  30732
 For    540 Secs 00:09:00                 YTU CSE DEPT. VM/ESA V2.2       ES/VM    20.02 SLU 0000
```

Chan in Dec	Chan in Hex	Pct Chan Busy	Channel Busy Percent									
			Busy =0	0< Busy <=10	10< Busy <=20	20< Busy <=30	30< Busy <=40	40< Busy <=50	50< Busy <=60	60< Busy <=70	70< Busy <=80	80< Busy <=90	90< Busy <=100
5	(05)	31.3	0	0	11	56	0	33	0	0	0	0	0
8	(08)	28.0	0	0	22	44	22	11	0	0	0	0	0
4	(04)	10.0	0	56	44	0	0	0	0	0	0	0	0
9	(09)	9.4	0	56	44	0	0	0	0	0	0	0	0

Tablo 7.6 Disk birimleri arasında giriş_çıkış işlemi sırasında disk aktivitesinin dağılımı

```
PRF012  Run 12/16/1999 22:00:50          DASD_BY_ACTIVITY                                    Page    2
                                          DASD Activity Ordered by Activity
 From 12/16/1999 21:50:31                                                                    TRYILDIZ
 To   12/16/1999 21:59:31                                                    CPU 9121        SN  30732
 For    540 Secs 00:09:00                 YTU CSE DEPT. VM/ESA V2.2          ES/VM     20.02 SLU 0000
```

<-----Device------>					<--SSCH+RSCH-->			<----------Time---------->					<-SSCHs in Queue-->		
Number	Volume Serial	Type	Control Unit	Owner	Mini-disk Links	On-line Secs	Count	Rate	Pct Busy	Pend	Disc	Conn	Serv	Resp	Mean	Max	Err
0117	ESAW07	3390-1	3990*EC		8	540	15992	29.6	94.1	0.8	20.6	10.3	31.8	31.8	0	0	0
0115	ESAW05	3390-1	3990*EC		7	540	15979	29.6	93.9	0.8	20.6	10.3	31.7	31.7	0	0	0
0101	ESAW01	3390-2	3990*EC		184	540	1167	2.2	4.0	0.4	2.9	15.0	18.3	18.3	0	0	0
0102	ESAW02	3390-2	3990*EC		153	540	103	0.2	0.2	0.4	5.0	2.7	8.1	8.1	0	0	0
0100	ESAW00	3390-2	3990*EC		147	540	95	0.2	0.1	0.4	4.4	2.9	7.7	18.2	0.0	0.0	0
0114	ESAW04	3390-1	3990*EC		1	540	19	0.0	0.0	0.3	0.0	0.9	1.2	1.2	0	0	0
0103	ESAW03	3390-2	3990*EC		79	540	18	0.0	0.0	0.3	0.0	0.9	1.2	1.2	0	0	0
0116	ESAW06	3390-1	3990*EC		2	540	18	0.0	0.0	0.3	0.0	0.8	1.2	1.2	0	0	0
0118	ESAW08	3390-1	3990*EC		2	540	18	0.0	0.0	0.3	0.0	0.9	1.2	1.2	0	0	0
0119	XXX059	3390-1	3990*EC		0	540	18	0.0	0.0	0.3	0.0	0.8	1.2	1.2	0	0	0
011A	XXX05A	3390-1	3990*EC		0	540	18	0.0	0.0	0.3	0.0	0.8	1.2	1.2	0	0	0
011B	XXX05B	3390-1	3990*EC		0	540	18	0.0	0.0	0.3	0.0	0.8	1.2	1.2	0	0	0
0The table above contains the top					12 of	12 items.											
Sum/Mean					49	6480	33463	5.2	16.0	0.8	19.8	10.4	31.0	31.1	0.0	0.0	0

8. SONUÇ

Yapılan bu doktora çalışmasında, ana bilgisayar sistemlerindeki giriş_çıkış altsisteminin, kişisel bilgisayar sistemlerindeki PCI yerel yolu ile bağlantısına ve PCI uyumlu çevre birimlerinin ana bilgisayar sistemlerine kullandırılmasına yönelik bir sistem geliştirilmiştir.

Bu tez çalışmasının özelliği, son yıllarda önem kazanan, ana bilgisayar sistemleri ile kişisel bilgisayar sistemlerinin entegrasyonunu, istemci_sunucu modeli kapsamında uygulanan ortak haberleşme protokollerine ve uygulamalara dayalı yapılara alternatif olarak, bu sistemler arasında giriş_çıkış altyapılarının donanım ile birleştirilmesiyle fiziksel seviyede gerçekleştirmesi ve kaynak paylaşımını sağlamasıdır. Bu çalışması kapsamında birbirleriyle fiziksel, elektriksel ve protokol özellikleri açısından farklılıklar gösteren ana bilgisayar sistemlerindeki paralel kanal giriş_çıkış altsistemi ile kişisel bilgisayar sistemlerindeki PCI yerel yolunun entegrasyonu için yazılıma ve donanıma dayalı bir model tarif edilmiş ve tariflenen model gerçeklenmiştir.

Bu konuda geliştirilmiş benzer sistemler incelendiğinde, kişisel bilgisayarlar için MicroChannel mimarisi kullanılarak geliştirilen özel birimler ile ana bilgisayar sistemlerine bağlantı sağlandığı ve bu bağlantı üzerinden veri aktarım hızı düşük olan teyp veya yazıcı gibi sınırlı tipte çevre biriminin desteklendiği görülmüştür. Bu sistemler, ana bilgisayar sistemine kullandırılacak çevre biriminin yapısına özel olarak geliştirildikleri için, belirlenmiş çevre biriminin kullanıcı tarafından değiştirilmesine imkan vermemektedirler. Bunun yanısıra mevcut sistemlerin, ana bilgisayar sistemine genellikle tek bir giriş_çıkış birimini kullandırdıkları, aynı tipten bile olsa ek birim kullanılmasına izin vermedikleri de görülmüştür.

MicroChannel mimarisi, kişisel bilgisayar giriş-çıkış işlemlerinde karşılaşılan darboğazı gidermek ve günümüzün gelişmiş yerel yol mantığına yakın bir yapıda, merkezi işlemciden bağımsız giriş_çıkış yapabilen birimlere de yer vermek amacı ile geliştirilmesine rağmen özel bir tasarım olması sonucunda standart olarak kabul görmemiştir. Kişisel bilgisayar çevre birimlerinin ve bunları kontrol eden birimlerin teknolojisindeki hızlı gelişme, veri

aktarım hızı, otomatik konfigürasyon gibi özellikler açısından MicroChannel mimarisinin yetersiz kalmasına sebep olmuş ve bu mimarinin kullanımı azalmıştır.

Bu doktora çalışmasında önerilen ve gerçekleştirilen sistem, ana bilgisayar sistemlerine kişisel bilgisayar çevre birimlerinin kullandırılması için mevcut sistemlerden farklı ve hem performans hem de içerdiği fonksiyonlar açısından yeni ve özgün bir yaklaşım getirmektedir.

Gerçekleştirilen sistemde, kişisel bilgisayar alanında endüstri standartı olan, yüksek hızlı veri aktarımı ve giriş_çıkış işlerinin işlemciden bağımsız yapılabilmesi özelliklerine sahip PCI yerel yolu kullanılmış, ana bilgisayar sistemine tanıtılacak PCI çevre birimlerinin tipinde bir kısıtlama getirilmemiş, aynı anda birden fazla farklı tipte çevre birimi desteklenmiştir. Bu özelliklere ve yeniliklere ek olarak, hata durumlarına karşı sistemin çalışmasını durdurmadan önlem alınabilmesi ve verilerin tek bir yapı altında kolayca erişilebilir şekilde saklanmasını (dataware housing) da desteklemek amacı ile sisteme yönettiği kaynaklar arasında dinamik atama yapabilme özelliği de kazandırılmıştır. Dinamik atama sayesinde ana bilgisayar sisteminin çevre birimine erişimi, PCI tarafında aynı tipte bir birimde gerçekleştirilebileceği gibi, yapı ve fonksiyon olarak benzer bir birime de yönlendirilebilmektedir.

Sistem tasarımında, ana bilgisayar sistemine ait paralel kanal yapısı ve PCI yerel yolu mimarisinin incelenmesi sonunda, veri haberleşmesinde farklı topolojiye ve protokole sahip yapılar arasında geçişin sağlanmasına benzer bir yaklaşımla katman modelinden yararlanılmıştır. Sistemde olması istenen özellikler ve fonksiyonlar katman yapısına göre değerlendirilip sınıflandırıldığında 4 katmana ihtiyaç olduğu belirlenmiştir. Bu katmanlar sırası ile **fiziksel katman, protokol katmanı, dönüşüm katmanı** ve **uygulama katmanı** olarak isimlendirilmiştir. Dönüşüm katmanı ve uygulama katmanı için kuyruk modelleri oluşturulmuştur. Katmanların gerçeklenmesinde, donanım ve yazılımlar yapılmıştır.

Fiziksel katman, gerçekleştirilen sistemin, paralel kanal ve PCI yerel yolu ile elektriksel bağlantısını sağlar. Protokol katmanı, fiziksel katmandan gelen sinyalleri değerlendiren ve bu sinyalleri anlamlı hale getiren katmandır. Dönüşüm katmanı, kişisel bilgisayar

sistemlerindeki giriş_çıkış birimleri ile ana bilgisayar sistemlerindeki giriş_çıkış birimleri arasındaki yapısal ve fonksiyonel farklılıkları ortadan kaldırmak amacıyla gerçeklenmiştir. Uygulama katmanı, dönüşüm katmanından gelen giriş_çıkış isteklerini, kişisel bilgisayar sisteminde ilgili çevre birimlerinde başlatmak, yürütmek, sonlandırmak ve dönüşüm katmanının bu çevre birimlerinin iç yapılarından bağımsız çalışabilmesini sağlamak amacıyla gerçekleştirilmiştir.

Ana bilgisayar sistemlerindeki paralel kanal ile kanal protokolüne uygun olarak fiziksel bağlantıyı kurmak, veri aktarımını gerçekleştirmek ve benzeri işlemleri yürütmek amacıyla, bu tez çalışması kapsamında, "Parallel Channel Interface Processor" olarak isimlendirilen özel bir birim tasarlanmış ve gerçeklenmiştir.

Bu birimin gerçeklenmesinde programlanabilir donanımdan yararlanılmıştır. Tasarımda yapılan değişikliklerin kolayca gerçekleştirilen sisteme aktarılabilmesi ve sistemin geliştirme zamanının azaltılması için 'In-system programming' yöntemi uygulanmıştır. Programlanabilir donanım birimlerinin içeriği, sistemin çalışması sırasında 'run-time reconfiguration' yöntemiyle değiştirilmiş ve aynı birimin farklı görevleri yerine getirebilmesi sağlanmıştır. 'Run-time reconfiguration' için JTAG arabirimini kullanan yeni ve özgün bir yöntem geliştirilmiş ve başarıyla çalıştığı gösterilmiştir.

Bu tez çalışmasında, ana bilgisayar sistemlerindeki paralel kanal yapısının PCI yerel yoluna bağlanması için önerilen sistemin tasarım ve gerçekleştirilmesinde donanımın yanında donanım birimlerini kontrol etmek amacı ile yazılımlar da geliştirilmiştir.

Protokol ve dönüşüm katmanına ait yazılım modüllerinin çalıştırılması, bu modüller arasında senkronizasyonun sağlanması, donanım ve yazılım kaynaklarına erişimin kontrol edilmesi ve bu kaynakların paylaştırılması amacı ile servis işlemcisinde çalıştırılmak üzere, gerçek-zamanlı çalışmaya uygun, çoklu programlama (multiprogramming) ortamını destekleyen, "Service Processor Control Program" (**SPCP**) adı verilen özel bir kontrol programı gerçekleştirilmiştir.

KAYNAKLAR

Alexandridis, N. A., (1984), "Microprocessor System Design Concepts", Computer Science Press, Inc., USA

Altera, "Altera Support for In-System Programmability"
http://www.altera.com/html/mktg/isp.html

Altera, "ISP via an Embedded Processor"
http://www.altera.com/html/mktg/isp-ep.html

Altera, "Device Selector Guide: Step 1 – Discover the Advantages of PLDs"
http://www.altera.com/html/products/dev-step1.html

Astrom, K., Wittenmark, B., (1990), "Computer Controlled Systems", Prentice Hall, Inc., New Jersey

Baer, J., (1980), "Computer Systems Architecture", Computer Science Press, Maryland

Comer, D., (1987), "Operating System Design-Volume II", Prentice Hall, Inc., New Jersey

Gartner, 1998, "Sunucu Sistemlerinde Sağlanabilirlik"

Gehani, N., McGettrick, A. D., (1988), "Concurrent Programming", AT&T Bell Laboratories

Hallsall, F., (1988), "Data Communications, Computer Networks and OSI", Addison Wesley Publishing Company, England

IBM, (1991), "Enterprise Systems Architecture/390, Channel-to-Channel Adapter for the System/360 and System/370 I/O Interface", IBM Corporation Enterprise Systems Central Architecture Department, USA

IBM, (1995), "Enterprise Systems Architecture/390, Principles of Operation", IBM Corporation Enterprise Systems Central Architecture Department, USA

IBM, (1992), "Enterprise Systems Architecture/390, System/360 and System/370 I/O Interface Channel to Control Unit Original Equipment Manufacturers' Information", IBM Corporation Enterprise Systems Central Architecture Department, USA

IDC, (1998), "Asset Management Services: Servers and Workstations, Bulletin, Server Selection: Reversing the Trend of Rising IT Costs", IDC #17428R, Framingham, MA 01701

Intel, (1994), "i960 CA/CF Microprocessor User's Manual", Intel Corporation, Literature Sale, USA

Intel, (1997), "i960 Processor Assembler User's Manual", Intel Corporation, Literature Distribution Center, USA

Intel, (1997), "PCI Efficient Use", Intel Corporation, Platform Architecture Labs

Jackson's Theorem,
http://www.treasure-troves.com/math/JacksonsTheorem.html

Jeruchim, M. C., Balaban, P., Shanmugan, K. S., (1994), "Simulation of Communication Systems", Plenum Press, New York

Levi, S., Agrawala, A. K., (1990), "Real Time System Design", McGraw-Hill, Inc., USA

Madnick, S., Donovan, J., (1978), "Operating Systems", McGraw-Hill, New York

Markov Equation, Markov Process
http://www.mth.kcl.ac.uk/~psollich/StatMechNN/node31.html

Miller, I., Freund, J., (1985), "Probability and Statistics for Engineers", Prentice Hall, Inc., New Jersey

Multi-Threaded Programming With POSIX Threads, http://www.actcom.co.il/~choo/lupg/tutorials/multi-thread/multi-thread.html

Navabi, Z., (1998), "VHDL, Analysis and Modelling of Digital Systems", McGraw-Hill, New York

OptiMagic, (1999), "Frequently-Asked Questions (FAQ) About Programmable Logic", OptiMagic Inc.

Papadimitriou, C., Steiglitz, K., (1998), "Combinatorial Optimization", Dover Publications, Inc., New York

PCI Bus Interface and Clock Distribution Chips Product Catalog, (1996), PLX

PCI Special Interest Group, (1995), "PCI Local Bus Specification", PCI Local Bus, Portland

Philippe, B., Saad, Y., Stewart, W., (1992), "Numerical Methods in Markov Chain Modelling", Operations Research, Vol. 40, No. 6, pp. 1156-1179

Re-programmable Hardware, http://www.dcs.gla.ac.uk/~jonathan/research/1styr/node7.html

Rosen, K., (1998), "Discrete Mathematics and Its Applications", McGraw-Hill, New York

Stallings, W., (1991), "Data and Computer Communications", Macmillan Publishing Company, New York

Tanenbaum, A., (1988), "Computer Networks", Prentice Hall, Inc., New Jersey

Tredennick, N., (1987), "Microprocessor Logic Design", Digital Press, USA

Wakerly, F. J., (1989), "Microprocessor Architecture and Programming", The 68000 Family", John Wiley & Sons, Inc., New York

ÖZGEÇMİŞ

Soyadı	YAVUZ
Adı	Ali Gökhan
Doğum Tarihi	7 Temmuz 1969
Doğum Yeri	İstanbul

Eğitim

1979-1986	İstanbul Erkek Lisesi
1986-1990	Yıldız Üniversitesi, Mühendislik Fakültesi Bilgisayar Bilimleri ve Mühendisliği Bölümü (Bilgisayar Mühendisi)
1991-1994	Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Bilgisayar Bilimleri Mühendisliği Anabilim Dalı (Bilgisayar Bilimleri Yüksek Mühendisi)
1994-1999	Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Bilgisayar Bilimleri Mühendisliği Anabilim Dalı

Yabancı Dil

İngilizce, Almanca

İş Deneyimleri

1991-	Araştırma Görevlisi Yıldız Teknik Üniversitesi, Elektrik-Elektronik Fakültesi Bilgisayar Bilimleri ve Mühendisliği Bölümü