LİNUX TABANLI AÇIK KAYNAK SES KARTI SÜRÜCÜSÜ GELİŞTİRİLMESİ

Kemal GÜVENLİ

Yüksek Lisans Tezi

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Ekim – 2007

LİNUX TABANLI AÇIK KAYNAK SES KARTI SÜRÜCÜSÜ GELİŞTİRİLMESİ

Kemal GÜVENLİ

Dumlupınar Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Elektrik-Elektronik Mühendisliği Anabilim Dalında
YÜKSEK LİSANS TEZİ
Olarak Hazırlanmıştır.

Danışman : Yrd. Doç. Dr. Serdar TUNABOYLU

Ekim - 2007

KABUL ve ONAY SAYFASI

Kemal GÜVENLİ 'nin YÜKSEK LİSANS tezi olarak hazırladığı Linux Tabanlı Açık Kaynak Ses Kartı Sürücüsü Geliştirilmesi başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir**.**

01/10/2007

Üye : Yrd. Doç. Dr. N. Serdar TUNABOYLU

Üye : Doç. Dr. Ahmet ALTUNCU

Üye : Yrd. Doç. Dr. Alpaslan DUYSAK

Fen Bilimleri Enstitüsün Yönetim Kurulu'nun ...../...../..... gün ve ..... sayılı kararıyla onaylanmıştır.

Fen Bilimleri Enstitüsü Müdürü
Prof. Dr. M. Sabri ÖZYURT

LİNUX TABANLI AÇIK KAYNAK SES KARTI SÜRÜCÜSÜGELİŞTİRİLMESİ

Kemal GÜVENLİ
Elektrik Elektronik Mühendisliği, Yüksek Lisans Tezi, 2007
Tez Danışmanı: Yrd. Doç. Dr. N. Serdar TUNABOYLU

ÖZET

Bu çalışmada, Linux işletim sisteminde atıl durumda olan bir ses kartı için sürücü geliştirilmiştir. Sürücü, çekirdek modülü olarak yazılmıştır. Bu modül, Linux çekirdeğine yüklenip kaldırılabilmektedir. Veri, ISA veri yolu üzerinden transfer edilmiştir.

Anahtar Kelimeler: Çekirdek Alanı, GNU, ISA veri yolu Kullanıcı Alanı, Linux, MED 3201, Modül

DEVELOPMENT OF A LINUX BASED OPEN SOURCE CODE OF A DRIVER

FOR AN AUDIO CARD

Kemal GÜVENLİ
Electrical and Electronics Department, Master Thesis, 2007
Thesis Supervisor: Assist. Prof. Dr. N. Serdar TUNABOYLU

SUMMARY

In this study, a driver has been developed for an audio card which is out of shelf for Linux operating system. The driver has been written as a kernel module. This module can be loaded to and removed from the Linux kernel. The data is transferred on ISA bus.
Keywords: GNU, ISA bus, Kernel Space, Linux, MED 3201, Module, User Space

TEŞEKKÜR

Bu çalışmada bana yardımcı olan danışman hocam Yrd. Doç. Dr. N. Serdar TUNABOYLU, hocalarım Yrd. Doç. Dr. Ahmet ÖZMEN, Yrd. Doç. Dr. Alpaslan DUYSAK, desteklerini hep yanımda hissettiğim ailem, Y. IŞIK, F. YÜCEL ve emeği geçen herkese teşekkürü bir borç bilirim.

İÇİNDEKİLER

İÇİNDEKİLER (devam)

Sayfa

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

Günümüzde kullanılan modern bilgisayar sistemleri, donanım, işletim sistemi ve uygulama yazılımları katmanlarından oluşmaktadır. İşletim sistemi bilgisayarı oluşturan donanımsal kaynakları (disk, bellek, ekran, yazıcı gibi) kullanıcılarına ve uygulama yazılımlarına kullandıran bir sistem yazılımıdır.

İşletim sistemleri aynı zamanda çeşitli yerlerden gelebilecek kötü amaçlı donanımsal erişim tehditlerini engelleyerek bilgisayar sisteminin sağlıklı bir şekilde çalışmasını da sağlarlar. Bunun ötesinde işletim sisteminin en önemli görevi, kullanıcı programlarının (uygulama yazılımları) donanımda gerçeklenmesi gereken işlemlerine sistem çağrıları yoluyla aracılık etmektir. Modern işletim sistemlerinde güvenlik açısından bu bir zorunluluktur. Bu yolla, yetkisiz veya kötü amaçlı erişimler işletim sistemi aracılığıyla engellenmiş olmaktadır.

Modern işletim sistemin başlangıcı, 1965 yılında duyurulan MULTICS (Multiplexed Information and Computer Service) kabul edilmektedir. Gelişimine MIT, GE ve AT&T Bell laboratuarları destek vermiştir. Projeye model olarak elektrik dağıtım şebekesini alındı. Elektrik ihtiyacı duyulduğunda fişin prize takılarak giderilmesi gibi eşzamanlı olarak yüzlerce kullanıcıyı bilgisayar donanımı desteklemeliydi. Projede istenilen noktaya ulaşılamadı ve askıya alındı [1].

1969 yılında Ken Thompson MULTICS projesine, kullanılmayan PDP-11 bilgisayarı üzerinde devam etti ve UNIX işletim sistemini ortaya koydu. UNIX, çok kullanıcılı ve çok görevli işletim sistemiydi. UNIX, C Programlama Dili'nin geliştirilmesinden sonra C kaynak koduyla yeniden yazılarak taşınabilir özelliği kazandı.

1991 yılında Linus Torvalds, Intel platformunda çalışan UNIX benzeri açık kaynak Linux çekirdeğini yayınladı. Linux'un açık kaynak yayınlanması bilgisayar dünyasında büyük ilgi gördü ve çoğu platformlara (DEC Alpha, Sun SPARC, Macintosh, AMD vb.) uyarlandı.

Linux aygıt sürücüleri, Linux'un platformlardaki aygıtları tanıyabilmesi için çekirdeğinin içine yerleştirilmektedir. Linux, her eklenen aygıt sürücülerle daha fazla donanımı destekler hale gelmektedir. Çoğu donanım üreten firma, kendi eski ürünleri için Linux sürücüsü yayınlamayı ekonomik bulmamakta ve sürücü yazmamaktadırlar. Linux gönüllüleri, üretici firması tarafından sürücüsü yazılmayan ürünleri atıl halden kurtarmak için kendi çabalarıyla sürücü yazmaktadırlar.

Bu çalışmada, AMD yonga setli MED 3201 Versiyon 1.2 model ses kartının Linux işletim sistemi için sürücüsü bulunamadığından, bir sürücü geliştirilmiştir. Geliştirilen sürücü ile kullanıcı programlarından sistem çağrıları yardımıyla veri transfer etmek mümkündür.

2. ÖZGÜR YAZILIM

2.1 GNU (Gnu Is Not Unix) ve Özgürlük

GNU Projesi, 1983 yılında MIT' den ayrılan yazılımcı Richard Stallman tarafından duyuruldu. 1985 yılında da GNU Manifestosu'nu yayınladı. GNU Projesi, özgür bir işletim sistemi oluşturmak için hazırlanmış başladı.

GNU Projesinin içinde çekirdeği GNU Hurd (henüz tamamlanamamıştır), kütüphanesi glibc, son kullanıcı programları (Emacs, gcc, gdb vb.) barındırır. Bu yazılımları kullanmak, dağıtmak, incelemek ve değiştirmek serbesttir. Bilgisayar dünyasında bir devrime yol açmış ve bilgisayar terminolojisine "**copyleft**" olarak girmiştir [1].

GNU Hurd çekirdeğinin tam olarak bitmemesinden dolayı Linux çekirdeğine ağırlık verilmiş, bütün uygulamaların Linux uyumluluğu sağlanmıştır.

Ek.1'de yer verilen Genel Kamu Lisansı(GPL), yazılımların "özgür" olmasını sağladı. Linux, C programlama dili ve GNU araçları kullanılarak yazıldı. Genel Kamu Lisansı ile lisanslandı. Gelişiminin hızını da bu lisansa borçludur.

"Özgür yazılım" özgürlükleri korumaya yönelik bir akımın adıdır. Temelinde kullanıcının bir yazılımı kopyalama, çalıştırma, inceleme, değiştirme, dağıtma, geliştirme özgürlüklerini sunar. Programın nasıl çalıştığını anlamak ve gereksinimler için değiştirebilme özelliği için kaynak koda erişim bir ön şart olmuştur.

Bir program, bu hakların tamamına sahip olması durumunda özgür yazılım olarak nitelenebilir. Bu hakları elde edebilmek için izin almaya gerek yoktur.

Özgür yazılımlar hakkında en sık rastlanan yanlış anlama, özgür yazılımların ücretsiz olmasıdır. Genel olarak ücretsiz olmasına karşın, özgür yazılım ücretli de olabilmektedir. Kaynak kodlarını yayınlamak yazılımı ücretlendirmekten bağımsızdır. Özgür yazılımlar kullanıcılarına şu temel hakları sağlarlar [2].

- Yazılımı kullanan kişi her türlü amaç için yazılımı çalıştırmakta özgürdür. Kullanıcı hakları özgür yazılımlar tarafından kısıtlanmazlar.
- Yazılımı kullanan kişi yazılımın çalışma biçimini incelemek ve kendi ihtiyaçları için yazılım üzerinde değişiklik yapmak hususunda özgürdür. Bu çalışmayı yardım alarak da yapabilir.
- Yazılımı kullanan elindeki yazılımı toplum ile paylaşmakta özgürdür. Dilediği kadar kopyasını dağıtabilir.
- Yazılımı kullanan kişi yazılımı geliştirmekte, geliştirdiği haliyle yazılımı toplum ile paylaşmakta özgürdür.

Özgür yazılım akımıyla, GPL altında geliştirilen bir yazılım ekstra çaba sarf etmeden, en başından en sonuna kadar insanlığın yararına geliştirilmiş olur. Bu lisans yaklaşımı sayesinde:

- Programcılar aynı şeyleri yeniden keşfetmek zorunda kalmaz, daha önceden üretilmiş olan araçları yazılımlarına ekleyerek daha yeni şeyler üretebilirler. Aşamayı bir basamak daha yukarı taşıyabilirler.
- Aynı işleri yapan yazılımların en iyisi hangisi ise onun özelliği alınıp, daha verimli yazılımlar üretilebilir. İhtiyaç duydukları kısımları diğer yazılımlardan alabilirler.
- Özgür yazılımlar genellikle ücretsizdir ama ücretli olanları makul fiyatlarla sunulmaktadır. Artan kaynaklar daha öncelikli alanlara kaydırılabilir.

Yazılımların kaynak kodunun açık olmasından dolayı, bütün açık kaynak yazılımların nasıl çalıştığının anlaşılmasının önündeki engel kaldırılmıştır. Bilgi transferi hızlı bir şekilde sağlanır. Toplumların bilgi seviyesi daha hızlı yükselir [3].

GPL sağladığı hükümler, kapalı kaynak kodunu savunan büyük yazılım devleri karşısında son kullanıcıları ve programcıları güvence altına almaktadır. Bu güvenceler ise şunlardır:

- **Güvenirlik:** Özgür yazılımların kaynak kodunun açık olmasında dolayı sağladığı en büyük güvence "güvenilirliktir". Kaynak kodu kapalı olan yazılımların aslında başka neler yaptığı tam kestirilememektedir. Açıkları kaynak kodu kapalı olduğu için tam giderilememektedir. Yüz binlerce göz tarafında incelenen açık kaynak yazılımların açıkları hızla giderilmektedir [4].
- **Sağlamlık:** Yazılım birçok kişi tarafından gözden geçirildiği için yüksek kaliteli ve kararlı yazılımlar ortaya çıkmaktadır.
- **Esneklik:** En hızlı şekilde açık kaynak kodlu yazılımlar yeni sistemlere entegre edilir.
- **Destek:** En büyük destek yazılımın açık kaynaklı olmasıdır. Daha sonra ise toplulukta sorunlar çözülmekte, çözümler aranmaktadır.

2.2 Linux

Linux, serbestçe dağıtılabilen, çok görevli, çok kullanıcılı, UNIX benzeri işletim sistemidir. Linux'un gelişimi internet üzerinde gönüllüler tarafından devam etmektedir. Birçok platformda çalışabilmektedir. Linux ücretsiz olduğu için son kullanıcıya bir maliyet getirmemektedir.

Linux, Finlandiya Üniversitesi Bilgisayar Bilimleri bölümünde yüksek lisans öğrencisi olan Linus Torvalds tarafından yazılmıştır. İnternet üzerinde birçok yazılımcının katkılarıyla gelişimini devam ettirmektedir. Gelişimi herkese açıktır. Her aşaması açık kaynak olarak www.linux.org sitesinde yayınlanmakta, çekirdek dünyanın dört bir yanında kullanıcılar tarafından test edilmekte, hataları

düzeltilmekte ve bu şekilde gelişimini sürdürmektedir. Belli zamanlarda kararlı çekirdek sürümleri yayınlanır ve geliştirme için ayrı bir seriye devam edilir. Bu Linux'un en büyük avantajıdır.

Linux ilk önce Linus Torvalds tarafından Minix üzerinde bir kullanıcı programı olarak tasarlandı. Minix'e ağ üzerinden bağlanan, veri alışverişini sağlayan bir istasyondu. 1991 yılında Linus tarafından daha da geliştirilerek çekirdek halini aldı. Linus, comp.os.minix haber grubuna bunu duyurdu ve herkesin desteğini beklediğini ekledi.

Çekirdek için verilen numaralar belli bir süre sonra standarta kavuştu. Çekirdek türevleri a.x.y şeklinde belirtilir. X gelişim aşamasını, y seviyeyi gösterir. Tek sayılı x'ler çekirdeğin hala geliştirilmekte olduğunu, çift sayılı x'ler ise çekirdeğin kararlı bir sürüme kavuştuğunu gösterir.

Linux'un dosya sistemi yapısında aşağıdaki dizinler yer alır:

/bin: Sistemin açılışı ve kontrolü için gerekli komutlar yeralır. Örnek olarak mkdir, cat, ps, su, insmod, chmod ve mount.

/dev: Çekirdek tarafından desteklenen her aygıta ait dosyalar bu dizinde yer alır.

/etc: Sistem yapılandırma dosyaları yer alır.

/lib: Kütüphane dosyaları yer alır.

/mnt: Mount edilen aygıt dosyaları yer alır. Fat32 dosya yapısına bu dizin altından erişilir.

/home: Kullanıcıya ait dizindir.

/proc: Sistem bilgileri ve süreç kontrol dosyalarının tutulduğu dizindir. Bu dizin çekirdek boot ettiğinde bellekte oluşturulur. İşlemci bilgileri, aygıtlar, kesme kullanım bilgileri ve giriş-çıkış iskele kulllanım bilgileri bu dizin altında saklanır.

/root: Sistem yöneticisinin dizinidir.

/tmp: Geçici dosyaların tutulduğu dizindir.

/usr: Diğer sistem dosyaları burada tutulur. Çekirdeğin kaynak kodları, kütüphaneler, yardım dosyaları, belge ve dökümanlar.

/var: Değişen sistem bilgileri burada tutulur. */var/log* dizininde çekirdeğin son bilgileri yer alır.

2.3 Kaynak Kodların Güvenirliği

Günümüzde işletim sistemleri tasarımında güvenlik ön planda tutulmaktadır. Mutlak güvenliğin olmadığı açıktır. En azından kapalı kaynak kodlu yazılımlara güvenmek tehlikelidir.

Ulusal güvenlik alanında devletler, bünyelerinde kullandıkları işletim sistemleri ve yazılımlarda kapalı kaynak kodlu yazılımlardan kesinlikle uzak durmalıdır.

Unix'in mucitlerinden Ken Thompson "Güvenilen Güven Üzerine Düşünceler " adlı ACM Ödül töreninde yaptığı konuşmada; Unix'in içine açıkkapı (backdoor) koyduğunu itiraf etmiştir. Bu açık kapı ile "ken" kullanıcı adıyla ve parolasız bir şekilde Unix sistemine girebildiğini açıklamıştır [4].

Ken Thompson; Unix'in açık kaynaklı olarak dağıtılmasına rağmen kompleks bir çalışmayla c derleyicini hatalı kaynak kodu derleyecek şekilde değiştirmiştir. Bu değişiklikleri ustaca göz önünden gizlemeyi başarmıştır. Bu noktada sorulması gereken açık kaynak kodlarının da ne kadar güvenilir olduğudur [4].

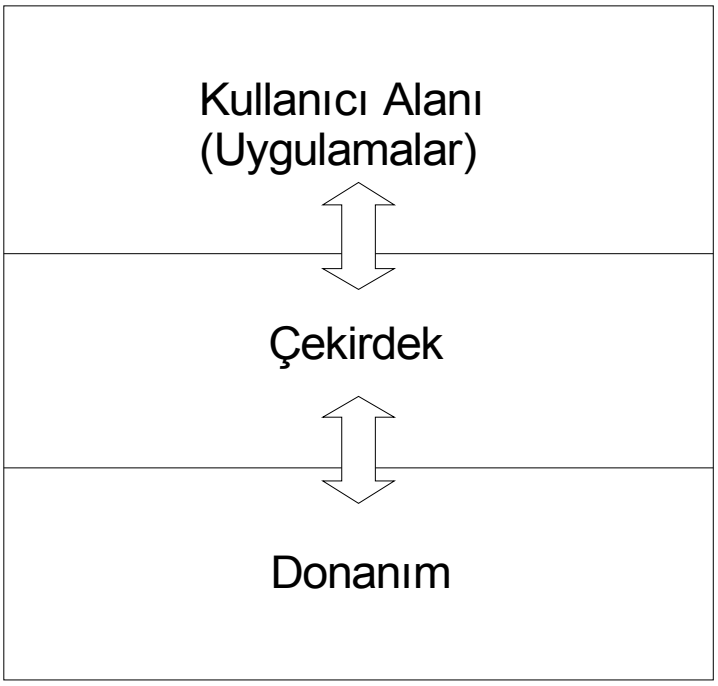
Kaynak kodları dünyaya açıksa -özgürse- bu kodlar milyonlarca programcı tarafından incelenebilmekte, test edilebilmekte ve geliştirilebilmektedir. Güvenlik hatalarını, eksikliklerini ve tasarımcısı tarafından hazırlanan açıkları incelemek açık kaynak kodlu yazılımlarda daha da mümkündür. Gerçek manada, derlenmiş kod (binary) geriye dönük olarak kaynak koda dönüştürülemediği için yazılımın kaynak koduna ulaşılamaz (reverse engineering), güvenlik testinden geçirilemez.

"Güvenirlik", bu çalışmada açık kaynak lisansın (GPL) tercih edilme sebeplerinden birisidir. Yaptığınız çalışmanın kaynak koduna güvenirliliğini açık kaynak kod lisansı önemli ölçüde artırmaktadır.

3. ÇEKİRDEK MODÜL PROGRAMLAMA

Linux çekirdeği kendi içindeki veri yapılarına ulaşan ve kendi içindeki rutinleri kullanan tek bir programdır. Bu yapıya monolitik çekirdek denir. Bu yapının alternatifi ise mikro çekirdek yapısıdır. Mikro çekirdek birbirleri arasında hataya az olanak verecek şekilde sıkı bir iletişim olan birçok programdan oluşan yapıdır. Bu programlar monolitik çekirdeğin fonksiyonerliğinin paylaştırıldığı bir yapı sunar. Monolitik çekirdeklere fonksiyonerlik eklemek için bütün yapıyı eklentiler ile baştan derlemek gerekir [5].

Şekil 3. 1 de çekirdek, donanım ve uygulamalar arasındaki etkileşim gösterilmiştir.



Şekil 3. 1 Donanım, çekirdek ve uygulamalar arasındaki etkileşim

GNU/Linux işletim sistemlerinin bir avantajı da çalışır sisteme çekirdek modülü ekleyebilme özelliğidir. Çalışan çekirdeğe gerektiği an ekstra özellikler eklenebilir, var olan özelliklerin davranışlarını kontrol edilebilir ve değiştirilebilir. Esneklik sağlayan bir modülerlik söz konusudur. İstenildiği zaman ve/veya gerekli olduğunda bu modüller çekirdek boot ettikten sonra bağlanabilir.

LKM (Yüklenebilir Çekirdek Modülleri) olarak bilinen modüller, Linux çekirdeğinin fonksiyonerliğini arttırmak için yazılan ek programcıklardır. Bu programcıklar sayesinde çalışan çekirdeği değiştirmek ve/veya eklentiler eklemek için yeniden derlemek gerekmez. Programları derlenerek çekirdeğe yüklenir ve program görevini yerine getirir. Örneğin bir ses kartı sürücüsü sonradan derlenip çekirdeğe yüklenerek ses kartın kullanılmasını sağlar [5].

Bir çekirdek modülü çekirdeğe bağlandığında artık çekirdeğin bir parçası olur. Dolayısı ile tıpkı çekirdekteki sorunlar gibi sistemin çakılmasına neden olabilir. Çekirdek kaynaklarını kullanan bu modüller bu kaynakları çekirdeğin kendi yapılarını yükleme aşamasında bulabilirler. Örneğin bir modül derlenme aşamasında **kmalloc()** fonksiyonu hakkında hiçbir şey bilmez. Bu bilgiler yükleme aşamasında çekirdeğin kendi içinde tuttuğu sembol tablosundan referansla düzenlenir.

Bir modül diğer modül yada modüllerin sembollerine ihtiyaç duyabilir. Bu durumda sembol kullanımına göre modüller sırası ile yüklenir, çekirdek yüklenen modülün gönderdiği sembolleri sembol tablosuna ekler, ardından gelen modül ise bu gönderilen sembolleri kullanır.

Modüller çıkartılırken çekirdek çıkartılan modülün kullanılmamakta olduğunu kontrol eder ve modülün kullandığı belleği ve/veya kesmeleri boşaltması için kendisine atanan çıkış fonksiyonunu işletir. Modül çıkarıldığında son olarak bu modülün gönderdiği semboller çekirdek sembol tablosundan çıkarılır ve çıkarma işlemi tamamlanmış olur.

Normal süreçler (proses) bir işi baştan sona yapmakla yükümlü gibidirler. Çekirdek Modülleri ise **init_module()** çağrısı ile kendi yapabileceklerini çekirdeğe bildirirler ve ana işlevleri biter. Bundan sonra çekirdek modülün kullanılacağı durumlarda modülün fonksiyonlarını kullanarak işlemleri gerçekleştirir.

Normal uygulamalar kendi tanımlamadıkları fonksiyonları bile ilgili kütüphanelerden kullanabilirler. Link verme aşamasında ilgili harici referanslar yapılır. Örneğin **printf, libc** içinde tanımlanmış çağırılabilir bir fonksiyondur. Fakat modüller çekirdeğe eklendiklerinden bu tür kütüphaneleri kullanamazlar. Bir modül yalnızca çekirdeğin gönderdiği fonksiyonları kullanabilir. Buna örneğe **printk()** fonksiyonu gösterilebilir [5].

Modüler normal kütüphaneleri kullanmadıkları için kaynak kodunda her zaman kullanılan kütüphane dosyaları yer almaz. Kullanabilecek kütüphane dosyaları "*/usr/src/include/linux*" ve "*/usr/src/include/asm*" dizinlerinde bulunur (Çekirdek "*/usr/src*" dizinine açılmalıdır). Kimi kullanıcı alanında çalışan programlar çekirdek kütüphanelerini de kullanabilirler. Bu kullanıcı alanında çalışan programların çekirdek kodlarını kullanmamaları için çekirdek kodlarının başına *#ifdef __KERNEL__* satırı eklenir.

insmod programı ile yüklemek istenilen obje dosyası (xxx.ko) yüklenilir, **create_module()** fonksiyonu ile modül hafızaya eklenecek, **printk** gibi çözülemeyen semboller **get_kernel_syms()** fonksiyonu ile çekirdek sembol tablosundan 'den öğrenilecek ve tanımlanan **init_module()** fonksiyonu çalıştırılır.

Sistem tarafından kullanılan bir modülün çekirdekten çıkartılması problemlere neden olur. Bunu engellemek için çekirdek kullanılan her modül için bir kullanım sayacı tutar. Böylece SCSI sabit diski kullanırken SCSI modülü çekirdekten çıkarılamaz. Hatalı yazılmış bir modül kullanıldığı anda

çıkarılabiliyorsa muhtemelen bu modülün sağladığı özellikleri kullanan program "segmentation fault" ile sonlanır yada bir "kernel panic" oluşur.

Yeni çekirdekler modülün kullanılıp kullanılmadığını algılayabilme özellikleri ile yazılırlar. Fakat eski çekirdekleri de göz önüne alarak basit olarak modül kullanım sayacı MOD_INC_USE_COUNT (sayaca 1 ekle) ve MOD_DEC_USE_COUNT (sayaçtan 1 çıkar) makroları ile değiştirilir. MOD_IN_USE, modül kullanılıyor ise doğru değer dönen bir makrodur. Bu makroları kullanarak modülün işlem yaptığı her alanda sayacı arttırmak ve her işlem bitiminde sayacı azaltmak gerekir. Sayaç sıfır ise modül çıkarılabilir.

Konsola bir satır yazı yazmaktan başka hiçbir işlevi olmayan bir modül, çekirdek ile 2 türlü iletişim kurabilir. Birincisi "**/dev**" dizininde oluşturulan aygıt dosyasını kullanarak. İkinci yöntem ise **"/proc**" dizininde oluşturulan bir dosya ile. **"/proc**" dizini özel bir dosya sistemidir. Bu dosya sistemine Sanal Dosya Sistemi (Virtual File System) denir. Çekirdek belleğinin bir bölümünün normal bir dosya sistemi olarak atandığı bu yapının içindeki dosyalar, çekirdek belleğine direk erişim sağlar.

İki tür aygıt türü vardır. Birincisi karakter aygıt tipidir (character device). Bu aygıt tipi stream veri okumak amaçlı kullanılır. Linux'ta konsollar bu tipe örnek olabilir. Diğer tip ise blok aygıtlardır (block device). Blok aygıtların çağrılar için bir tampon belleği vardır, dolayısıyla cevapları kendi istedikleri sıraya göre verebilirler. Bu özellik veri saklama aygıtlarında oldukça fazla yarar sağlar. Örneğin o anki sektöre yakın sektörleri önce yazmayı ya da okumayı seçebilmek gibi. Diğer bir özellik ise blok aygıtlar çağrıları ve cevapları belirli bir uzunluktaki bloklar halinde alırlar ve verirler. Bu uzunluk kullanılan aygıta göre değişir. Buna karşın karakter aygıtları istenilen uzunlukta çağrı alabilir ve cevap verebilirler.

Aşağıda **"/proc/devices**" dosyasındaki aygıtlar listelenmektedir. Yeni yazılan çekirdek modülü de bu listeye eklenir.

```
$ cat /proc/devices

Character devices:
  1 mem
  2 pty/m%d
  3 pty/s%d
  4 tts/%d
  5 cua/%d
  6 lp
  7 vcs
 10 misc
 13 input
 14 sound
```

36 netlink
108 ppp
109 lvm
116 alsa
128 ptm
136 pts/%d
162 raw
180 usb
195 nvidia

Block devices:

1 ramdisk
2 fd
3 ide0
7 loop
8 sd
...

4. SES KARTI

Bu çalışmada, AMD Interwave AMC200KC yonga set, Data Expert marka ve MED 3201 Ver1.2 model ses kartı için hazırlanan sürücüyle, GNU/Linux işletim sitemi altında çalıştırılarak atıl durumdan kurtarılması amaçlanmıştır.

Günümüzde firmalar, eski donanımları için güncel işletim sistemlerine sürücü geliştirmeyi ekonomik bulmadıklarından yapmamaktadırlar. Bu da sağlam eski donanımları güncel işletim sistemlerinde atıl duruma getirmektedir. Bu gelişmekte olan ülkeler için çok büyük bir ekonomik kayıp olmaktadır.

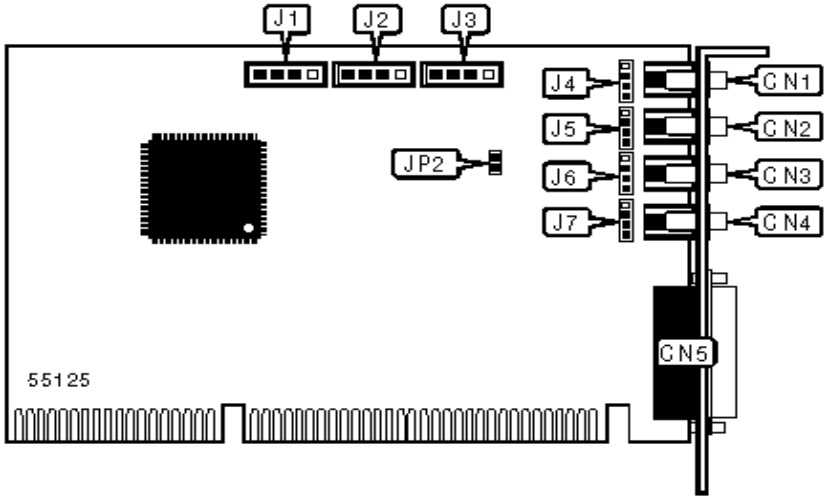
Son zamanlarda donanım üretici firmalar arasındaki eğilim; teknik desteklerinin çoğunu -bazıları için tamamını- pazar payı çok büyük olan Microsoft işletim sistemlerine vermek olmuştur. Ticari olarak çok karlı olan bu işletim sistemleri nedeniyle firmaların giderek özgür yazılım dünyasına olan destekleri azalmaktadır. Açık kaynak yazılım ürünü olan GNU/Linux işletim sistemlerinde bu durum çok büyük problem oluşturmaktadır. İşletim sistemi (GNU/Linux) için sürücüsüz bir donanımın plastik bir karttan farkı yoktur.

4.1 MED 3201

MED 3201, Data Expert firmasının ürettiği ISA veri yolunu kullanan 16-bit tak çalıştır bir ses kartıdır. Ses kartının özellikleri Tablo 4.1 de listelenmiştir. MED 3201 ses kartının resmi Şekil 4.1 de yer almış ve resimde gösterilen bağlantılar Tablo 4.2 de açıklanmıştır.

Tablo 4.1 Ses kartının özellikleri

Kart Tipi	Ses Kartı
Yonga Seti	AMD, AM78C200KC yonga set
I/O Opsiyonları	Bağlantı girişi (2), bağlantı çıkışı (2), ses çıkışı (2), mik. girişi (2), game/MIDI port, ses girişi - CD-ROM (3), PC hoparlör
Data Veri Yolu	16-bit ISA
Kart Boyutları	Yarım geniş, tam yüksek kart



Şekil 4.1 Med 3201

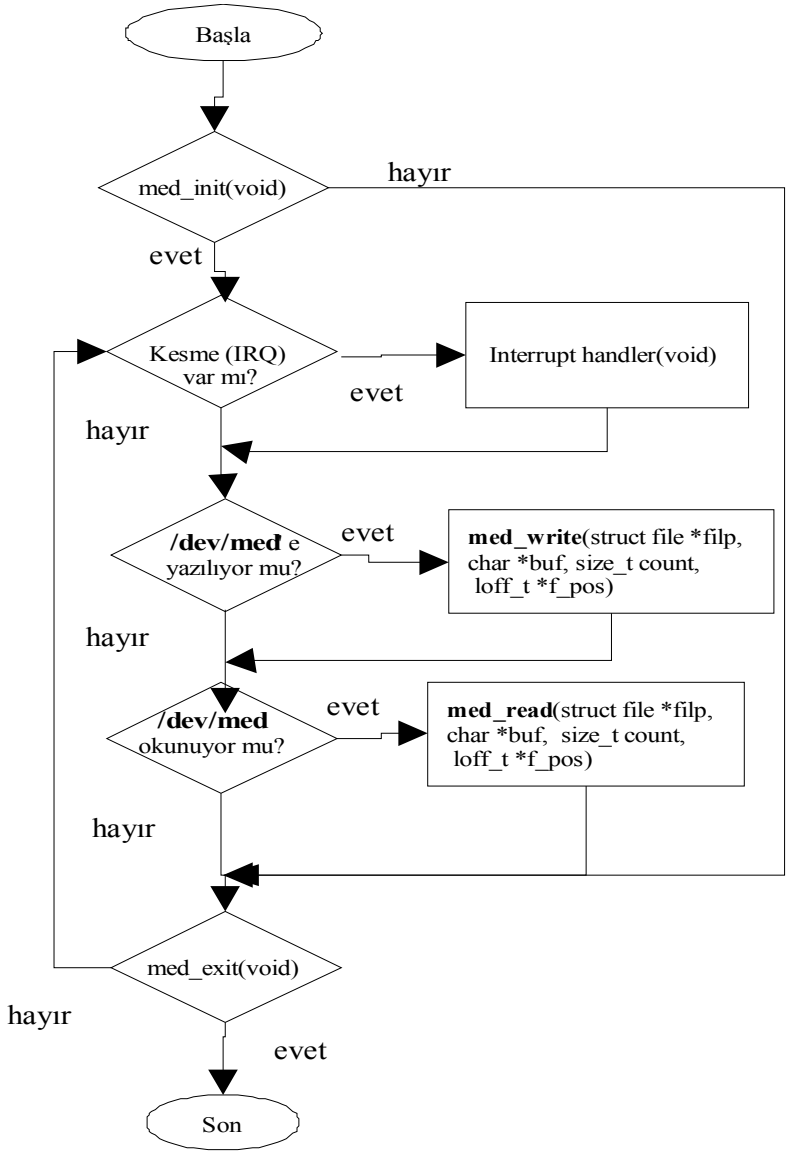
Tablo 4. 2 Bağlantılar

BAĞLANTILAR			
FONKSİYON	**Etiket**	**Fonksiyon**	**Etiket**
Hoparlör çıkışı - harici	CN1	Ses girişi - CD-ROM	J3
Bağlantı çıkışı – harici	CN2	Hoparlör çıkışı - dahili	J4
Ses girişi - harici	CN3	Bağlantı çıkışı - dahili	J5
Mikrofon girişi – harici	CN4	Bağlantı girişi - dahili	J6
Oyun/MIDI port	CN5	Mikrofon girişi - dahili	J7
Ses girişi - CD-ROM	J1	PC Hoparlör çıkışı	JP2
Ses girişi - CD-ROM	J2		

5. SES KARTI SÜRÜCÜSÜ

5.1 Yazılımın Algoritması

Ses kartı sürücüsünün algoritması Şekil 5.1'de gösterilmiştir. Algoritmanın açıklamasına 5. 2 başlığı altında yer verilmiştir.



Şekil 5. 1 Med 3201 model ses kartı sürücüsünün algoritması

5.2 Kaynak Kod Açıklaması

Öncelikle sürücü için gerekli olan kütüphane dosyaları deklare edildi. Kullanıcı programlarından farklı olarak buradaki kütüphane dosyaları çekirdek kütüphane dosyalarıdır. Sistem programlarında diğer kütüphane dosyaları kullanılamaz.

module_init(med_init);

module_exit(med_exit);

Bu fonksiyonlar modül çekirdeğe yükleneceği ve atılacağı zaman çalışan fonksiyonlardır. module_init() fonksiyonunda öncelikle aygıt kaydedilir (register_chrdev()). Bu:

 # mknod /dev/med c 67 0

komutuyla oluşturulan karakter dosyanın med.ko (med.c dosyasının derlenmiş halidir) dosyasıyla ilişkilendirildiği yerdir. Burada major numarasıyla aygıt çekirdek tarafından tanınabilmektedir.

request_irq() fonksiyonuyla donanımdan kesme gelip gelmediği kontrol edilir, eğer bir kesme varsa interrupt_handler() fonksiyonu icra edilir.

kmalloc() fonksiyonuyla çekirdek bellek alanında tampon tahsis edilir. memset() fonksiyonuyla tamponun içeriği istenen veriyle doldurulur.

check_region() fonksiyonuyla belirtilen iskelenin kullanımda olup olmadığı kontrol edilir, kullanımdaysa hata yakalama çalışır değilse o adres kullanıma açılır [6].

module_exit() fonksiyonunda modul_init() fonksiyonunun tam tersi uygulanır. Aygıt, tampon, iskele serbest bırakılır. Kesme engellenir.

Çekirdek alan fonksiyonu, kullanıcı alanındaki (**fopen**) açılan bir dosyayla uyuşan, **register_chrdev** çağırmada file_operations yapısının üyesi **open**: dır. bu durumda o **med_open** fonksiyonudur [7]. Argüman olarak: Çekirdeğe major ve minor numara hakkında bilgi yollayan **inode** yapısıdır; farklı işlemleri bir dosyada yapabilmeye ilişkin bilgiyle file yapısıdır.

Kullanıcı fonksiyonu **fread** le bir aygıtı okumak için file_operations yapısının üyesi **read: register_chedev** çağırmada kullanılır. Bu sefer fonksiyon **med_read** dir. Onun argümanları: bir çeşit dosya yapısı; tampon (buf), kullanıcı alanı fonksiyonu (**fread**) okuyacakları için; sayaç, transfer edilecek baytların sayısı için (count); sonunda dosyayı okumaya başladığı pozisyon (**f_pos**). Açıkça, **med_read()** fonksiyonu sürücü tamponundan (**med_buffer**) kullanıcı alanına fonksiyon **copy_to_user** ile tek bayt transfer eder.

Dosyada okuma pozisyonu (f_pos) de değiştirilir. Eğer pozisyon dosyanın başındaysa o bir arttırılır ve uygun şekilde okunan bayt sayısı geri dönüş değeri olarak 1 verir. Eğer dosyanın başında değilse dosyanın sonu (0) geri döner.

ISA veri yolundan veri okumak için **inb()** fonksiyonu kullanılır.

Kullanıcı fonksiyonu **fwrite** le bir aygıtı yazmak için file_operations yapısının üyesi **write: register_chedev** çağırmada kullanılır. Bu sefer fonksiyon **med_write** dir. Onun argümanları: bir çeşit dosya yapısı; tampon (buf), kullanıcı alanı fonksiyonu (**fwrite**) okuyacakları için; sayaç, transfer edilecek baytların sayısı için (count); sonunda dosyayı okumaya başladığı pozisyon (f_pos). Açıkça, **med_write()** fonksiyonu sürücü tamponundan (med_buffer) kullanıcı alanına fonksiyon **copy_from_user** ile tek bayt transfer eder [7].

ISA yeri yoluna veri yazmak için outb() fonksiyonu kullanılır.

Çekirdek modülde **printk** fonksiyonu printf fonksiyonuyla benzerdir. **Farklılığı sadece çekirdeğin içinde çalışıyor** olmasıdır**. <1> sembolü mesajın yüksek önceliğini gösterir. Diğer sembollerin anlamı Tablo 5.1 de gösterilmiştir.**

Tablo 5. 1: Printk sembolleri

<0>	sistem kararsız
<1>	acil önlem alınmalı
<2>	kritik koşul
<3>	hata koşulu
<4>	uyarı koşulu
<5>	normal fakat önemli durum
<6>	bilgi
<7>	debug-level mesaj

Böylece, çekirdek sistem log dosyaları içinde mesajı almanın yanında, mesaj sistem konsolunda da alınabilir. Bu modülün yüklenmesinde veya kaldırılmasın konsolda gözükür. Eğer gözükmezse:

#cat /var/sys/syslog

veya

#dmesg | less komutlarıyla çıktılara ulaşılabilir [8].

Tablo 5.2'de kullanıcı fonksiyonlarını çekirdek tarafında gerçekleyen fonksiyonlar listelenmiştir.

Tablo 5. 2: Çekirdek ve kullanıcı alanındaki aygıt sürücü olayları ve onların birleştirilmiş ara yüz fonksiyonları

Olaylar	Kullanıcı Fonksiyonları	Çekirdek fonksiyonları
Modül yükle (Load module)	insmod	module_init()
Aygıtı aç (Open device)	fopen	file_operations: open
Aygıtı oku (Read device)	fread	file_operations:read
Aygıta yaz (Write device)	fwrite	file_operations:write
Aygıtı kapat (Close device)	fclose	file_operations:release
Aygıtı kaldır (Remove devices)	rmmod	module_exit()

Ek. 2'de yer verilen kaynak kod hazırlanan bir Makefile ile derlenebilmektedir. Derleme sonucunda med.ko dosyası oluşur. Bu modül:

 # insmod med.ko

komutuyla belleğe yüklenir.

#lsmod

komutuyla bellekteki yüklü modüller listelenmektedir.

#rmmod med.ko

komutuyla da bellekten modül çıkartılır.

6. VERİ YOLLARI SİSTEMİ

Veri yolları işlemciyi bilgisayarın belleğine(RAM) ve yuvalara bağlar. Bilgiler bu veri yolları sayesinde transfer edilir. İşlemci bu sayede belleğe erişir. Veri iletiminde kullanılan bu bağlantılara "Veri Yolu" denir.

Veriler işlemci tarafından direk veri yoluna gönderilmez. Veriler gideceği yer belirtilerek gönderilir. Gideceği yer adres yolu denen bağlantı kümesi tarafından gerçekleştirilir. Sistem yolu vasıtasıyla da adres yolu üzerinde denetim sağlanır. Okuma, yazma ve adresleme işlemleri sistem yolu sayesinde basitleştirilir. Sistem yolu bu görevlerini yol denetleyicileri yardımıyla yapar. Yol denetleyicileri işlemlerin çarpışmasını, üst üste binmesini önler.

İşlemcinin veri genişliğinin arttırılamayacağından dolayı yol sistemin çalışma frekansı yükseltilerek hız arttırılabilir. Standart IBM AT yolu 8MHz hızla çalışır.

Genişleme yuvaları yol sistemine bağlı soketlerdir. Bu genişleme yuvalarına ekran kartı, ses kartı, ethernet kartı, usb diskler, sabit disk denetleyicileri vb. bağlanır. Genişleme yol sistemleri:

1. ISA (Industry Standart Architecture / Endüstri Standart Mimarisi):

ISA, "Industry Standart Architecture" teriminin kısaltılmasıdır. ISA veri yolu 16-bit AT veri yolu mimarisi ve 8-bit XT veri yolu mimarisi olarak ikiye ayrılır. Tablo 6. 1'de XT veri yolunun Tablo 6. 2'de de AT veri yolunun özellikleri yer almaktadır.

Tablo 6. 1: XT veri yolu

DMA Kanalı	Genişleme	Standart Fonksiyon
0	Hayır	Dinamik hafıza Yenileme
1	Evet	İlave kartlar
2	Evet	Disket Sürücü denetleyici
3	Evet	Sabit Disk denetleyici

Tablo 6. 2: AT veri yolu

Veri Yolu Genişliği	8-bit
Bağdaşabildiği	8 bit İSA
Pinler	62
Vcc	+5 V, -5 V, +12 V, -12 V
Saat Vuruşu	4.7727266 MHz

ISA, 8-bit veri yolu yanında 20-bitlik adres yoluna sahiptir. Ek. 3'de veri yolu açıklamalarıyla beraber gösterilmiştir. Gelişen yeni veri yolları nedeniyle bugünün standartlarından oldukça geri düşmüştür [9].

2. EISA (Enchanced Industry Standart Architecture / Geliştirilmiş Endüstri Standardı Mimarisi):

EISA, AT veri yolunun geliştirilmiş versiyonudur. Bu standart 32-bit işlemcilerin yüksek performans talebini karşılamak için oluşturuldu. Veri yolunun 32-bit olması nedeniyle işlemcinin 32 bitini aynı anda iletebilmekteydi. Teorik olarak saniyede 8 Megabit transfer yapabilmektedir. Pratikteyse 1 yada 2 Megabit hızında çalışabilmektedir.

Bunun yanında diğer önemli özelliği birden fazla işlemci veri yoluna bağlanabilmekteydi. Çoklu işlemciyi destekliyordu. ISA'ya uyumlu olduğu için ISA genişleme kartlarını da destekliyordu.

3. MCA (Micro Channel Architecture / Mikro Kanal Mimarisi):

Bu veri yolunda, veriler belirtilen adresle hedefine ulaştırılmaz. Veri yolu veriler belirtilen adreslerden çekilip alınacağı bir kanaldan oluşur. Önce verinin yeri bildirilir, sonra verinin aktarılacağı kanala erişim izni tanınır. Veri aktarım hızı 40 MBps, saat frekansı 10.33MHz'dir [10].

4. PCI (Peripheral Component Interconnect):

PCI, İntel tarafından geliştirilen an yaygın veri yoludur. 32 bit genişliğinde olmasına rağmen 64 bit olarak da çalışabilir. PCI, 33 MHz hızında çalışabilecek şekilde üretilmiştir. Veri yolu tamponlu çalışabilecek şekilde tasarlanmıştır. İşlemcinin görevleri tamponda biriktirilerek görevler sırasıyla yerine getirilir. Bu işlemin tersini de kullanır. PCI kartlar "tak çalıştır" özelliğine sahip olduklarından, PCI kartlar kendilerini konfigüre ederek sisteme tanıtırlar.

5. AGP (Accelerated Graphics Port / Hızlandırılmış Grafik Portu):

AGP, 66 MHz çalışan PCI veri yoludur. Ekran kartlarından daha hızlı ve gerçekçi görüntüler elde etmek için geliştirilmiş veri yoludur. İşlemciye en yakın veri yoludur.

3B grafikler, yüksek çözünürlükte ve hızlı hareket ettirildiğinde PCI veri yolunun sınırlarını zorladığı için AGP geliştirilmiştir. AGP'nin veri yolunun tüm bant genişliği sadece grafik için kullanılmaktadır.

7. SONUÇLAR

Bu çalışmada, C programlama dili kullanılarak, Linux çekirdeği üzerinde çalışacak bir ses kartı sürücüsü yazılmıştır. Geliştirilen bu sürücü ile kullanıcı programlarından ilgili ses kartına erişmek mümkün olmuştur. Kullanıcı programlarına ilave edilecek kütüphaneler ile ses kartının belleğine veri transferi yapılmış, ancak model olarak oldukça eski olan ses kartına ilişkin tanıtıcı dokümanlara ulaşılamadığından ses kartının çıkışından ses elde edilememiştir.

.

KAYNAKLAR DİZİNİ

[1] http://cekirdek.uludag.org.tr/~meren.

[2] Stallman M.R., Haziran 1991, Gnu Public License, Version 2, http://www.gnu.org/copyleft.

[3] http://www.gnu.org.

[4] Thompson K.,1984, Reflections on Trusting Trust, *C. of the ACM*, Vol. 27, No.8, 761-763 s.

[5] http://www.thehackerschoice.com/papers, Complete Linux Loadable Kernel Modules.

[6] Salzman P. J. Pomerantz O., 2001, The Linux Kernel Module Programming Guide.

[7] Carbet X. , *2006 ,* Writing device drivers in Linux: A brief tutorial, F. S. Magazine, issue 11.

[8] Corbet J., Rubini A., ve Kroah-Hartman G., Linux Device Driver.

[9] http://www.bote.ogu.edu.tr.

[10] http://www.geocities.com/matbaa19myo.

Ek 1. GNU Genel Kamu Lisansı

Sürüm 2	Haziran 1991	Telif Hakkı © 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Bu lisans dokümanının birebir kopyalarını yapma ve dağıtma izni herkese verilmiştir, fakat metinde değişiklik yapma izni yoktur.		

Yazılım lisanslarının çoğu sizin yazılımı paylaşma ve değiştirme hakkınızın elinizden alınması için hazırlanmıştır. Buna karşılık, GNU Genel Kamu Lisansı sizin serbest yazılımları değiştirme ve paylaşma hakkınızın mahfuz tutulması ve yazılımın bütün kullanıcıları için serbest olması amacı ile yazılmıştır. Bu Genel Kamu Lisansı, Free Software Foundation'un çoğu yazılımı ve bu lisansı kullanmayı düstur edinen diğer yazılımcıların yazılımları için kullanılmaktadır. (Free Software Foundation'un bazı yazılımları GNU Kısıtlı Genel Kamu Lisansı -- GNU LGPL -- altında dağıtılmaktadır.) Siz de bu lisansı yazılımlarınıza uygulayabilirsiniz.

Serbest yazılımdan bahsettiğimiz zaman fiyattan değil, özgürlükten bahsediyoruz. Bizim Genel Kamu Lisanslarımız, sizin serbest yazılımların kopyalarını dağıtma özgürlüğünüzü (ve isterseniz bu hizmet için para almanızı), yazılım kaynak kodlarının size dağıtım esnasında veya eğer isterseniz verilmesini, yazılımı değiştirebilmenizi, yazılımın parçalarını yeni yazılımlar içerisinde kullanabilmenizi ve bunları yapabileceğinizi bilmenizi sağlamaktadır. [3]

Haklarınızı koruyabilmemiz için sizin haklarınızı kısıtlama veya sizin bu haklarınızdan feragat etmenizi isteme yollarını yasaklayıcı bazı kısıtlamalar getirmemiz gerekmektedir. Bu kısıtlamalar eğer serbest yazılım dağıtıyor veya değiştiriyorsanız size bazı yükümlülükler getirmektedir.

Örneğin böyle bir programın kopyalarını, bedava veya ücret karşılığı dağıtıyorsanız alıcılara sizin sahip olduğunuz bütün hakları sağlamalısınız. Onların da kaynak kodlarına sahip olmalarını veya ulaşabilmelerini sağlamalısınız. Onlara da haklarını bilebilmeleri için bu şartları göstermelisiniz.

Haklarınızı iki koruma iki aşamada gerçekleşmektedir:

Yazılıma telif hakkı alınmaktadır.

Yazılım lisansı olarak size, hukuki olarak, yazılımı kopyalama, dağıtma ve/veya değiştirme hakkı tanıyan bu lisans sunulmaktadır.

Ayrıca, yazarların ve bizim korunmamız için bu serbest yazılımın herhangi bir garantisi olmadığını herkesin anlamasını istiyoruz. Eğer yazılım başkası tarafından değiştirilmiş ve değiştirilmiş hali ile tarafınıza ulaştırılmış ise alıcıların, ellerinde olan yazılımın orijinal olmadığını, dolayısıyla

başkaları tarafından eklenen problemlerin ilk yazarların şöhretlerine olumsuz etkide bulunmaması gerektiğini bilmelerini istiyoruz.

Son olarak, bütün serbest yazılımlar yazılım patentleri tarafından sürekli tehdit altında bulunmaktadır. Serbest bir yazılımın dağıtıcılarının bireysel olarak patent lisansı almalarını ve bu yol ile yazılımı müseccel hale getirmelerine imkan vermemek istiyoruz. Bunu engellemek için, yazılım için alınacak her patentin herkesin serbest kullanımına izin vermesi veya patentlenmemesi gerektiğini açık olarak ortaya koyuyoruz.

Kopyalama, dağıtım ve değiştirme ile ilgili kesin şart ve kayıtlar 2.1.3. başlığında yer almaktadır.

Kopyalama, Dağıtım ve Değiştirme İle İlgili Şart ve Kayıtlar

0. Bu Lisans, telif hakkı sahibi tarafından içerisine bu Genel Kamu Lisansı altında dağıtıldığına dair ibare konmuş olan herhangi bir yazılım veya başka eseri kapsamaktadır. Aşağıda "Yazılım", bu kapsamdaki herhangi bir yazılım veya eser, "Yazılımı baz alan ürün", ise Yazılım veya telif kanunu altında Yazılım'dan iştikak etmiş, yani Yazılım'ın tamamını veya bir parçasını, değiştirmeden veya değişiklikler ile veya başka bir dile tercüme edilmiş hali ile içeren herhangi bir ürün, manasında kullanılmaktadır. (Bundan sonra tercüme "değiştirme" kapsamında sınırsız olarak içerilecektir.) Her ruhsat sahibine "siz" olarak hitap edilmektedir.

Kopyalama, dağıtım ve değiştirme haricinde kalan faaliyetler bu Lisans'ın kapsamı dışındadırlar. Yazılım'ı çalıştırma eylemi sınırlandırılmamıştır ve Yazılım'ın çıktısı yalnızca çıktının içeriği (Yazılım'ı çalıştırmak yolu ile elde edilmesinden bağımsız olarak) Yazılım'ı baz alan ürün kapsamına girer ise bu Lisans kapsamındadır. Bu koşulun sağlanıp sağlanmadığı Yazılım'ın ne yaptığı ile ilgilidir.

1. Yazılım'ın kaynak kodlarını birebir, aldığınız şekilde, herhangi bir ortamda ve vasıta ile, uygun ve görünür bir şekilde telif hakkı bildirimi ve garantisiz olduğuna dair bildirim koymak, bu Lisans'dan bahseden herhangi bir bildirimi aynen muhafaza etmek ve bütün diğer alıcılara Yazılım ile birlikte bu Lisans'ın bir kopyasını vermek şartı ile kopyalayabilir ve dağıtabilirsiniz.

Kopyalamak fiili işlemi için bir ücret talep edebilir ve sizin seçiminize bağlı olarak ücret karşılığı garanti verebilirsiniz.

2. Yazılım'ın kopyasını veya kopyalarını veya herhangi bir parçasını değiştirerek Yazılım'ı baz alan ürün elde edebilir, bu değişiklikleri veya ürünün kendisini yukarıda 1. bölümdeki şartlar dahilinde ve aşağıda sıralanan şartların yerine getirilmesi koşulu ile kopyalayabilir ve dağıtabilirsiniz.

Değiştirilen dosyaların görünür bir şekilde dosyaların sizin tarafınızdan değiştirildiğine dair, tarihli bir bildirim içermesini sağlamalısınız.

Yazılım'dan veya Yazılım'ın bir parçasından tamamen veya kısmen iştikak etmiş ve sizin tarafınızdan dağıtılan veya yayınlanan herhangi bir ürünün bütün üçüncü şahıslara bu Lisans şartları altında ücretsiz olarak ruhsatlanmasını sağlamalısınız.

Eğer değiştirilen yazılım olağan kullanım altında komutları interaktif olarak alıyor ise, yazılım, en olağan kullanım için interaktif olarak çalıştırıldığı zaman uygun bir telif hakkı bildirimi, garantisi olmadığına (veya sizin tarafınızdan garanti verildiğine), kullanıcıların bu yazılımı bu şartlar altında tekrar dağıtabileceklerine ve kullanıcının bu Lisansın bir kopyasını nasıl görebileceğine dair bir bildirim yazdırmalı veya göstermelidir. (İstisna: Eğer Yazılım'ın kendisi interaktif ise fakat böyle bir bildirimi olağan kullanım esnasında yazdırmıyor ise, sizin Yazılım'ı baz alan ürününüz böyle bir bildirimde bulunmak zorunda değildir.)

Bu şartlar değiştirilmiş eserin tamamını kapsamaktadır. Eğer eserin tespit edilebilir kısımları Yazılım'dan iştikak etmemiş ise ve makul surette kendi başlarına bağımsız ve ayrı eserler olarak kabul edilebilir ise, o zaman bu Lisans ve şartları, bu parçaları ayrı eser olarak dağıttığınız zaman bağlayıcı değildir. Fakat, aynı parçaları Yazılım'ı baz alan bir ürün bütününün bir parçası olarak dağıttığınız zaman bütünün dağıtımı, diğer ruhsat sahiplerine verilen izinlerin bütüne ait olduğu ve parçalarına, yazarının kim olduğuna bakılmaksızın bütün parçalarına tek tek ve müşterek olarak uygulandığı bu Lisans şartlarına uygun olmalıdır.

Bu bölümün hedefi tamamen sizin tarafınızdan yazılan bir eser üzerinde hak iddia etmek veya sizin böyle bir eser üzerindeki haklarınıza muhalefet etmek değil, Yazılım'ı baz alan, Yazılım'dan iştikak etmiş veya müşterek olarak ortaya çıkarılmış eserlerin dağıtımını kontrol etme haklarını düzenlemektir.

Buna ek olarak, Yazılım'ı baz almayan herhangi bir ürünün Yazılım ile (veya Yazılım'ı baz alan bir ürün ile) bir bilgi saklama ortamında veya bir dağıtım ortamında beraber tutulması diğer eseri bu Lisans kapsamına sokmaz.

3. Yazılım'ı (veya 2. bölümde tanımlandığı hali ile onu baz alan bir ürünü) ara derlenmiş veya uygulama hali ile 1. ve 2. Bölüm'deki şartlar dahilinde ve aşağıda sıralanan yöntemlerden birisine uygun olarak kopyalayabilir ve dağıtabilirsiniz.

Yaygın olarak yazılım dağıtımında kullanılan bir ortam üzerinde, yukarıda 1. ve 2. Bölüm'de bulunan şartlar dahilinde, bilgisayar tarafından okunabilir kaynak kodlarının tamamı ile birlikte dağıtmak.

Herhangi bir üçüncü şahsa, fiziksel olarak dağıtımı gerçekleştirme masrafınızdan daha fazla ücret almayarak, yaygın olarak yazılım dağıtımında kullanılan bir ortam üzerinde, yukarıda 1. ve 2. Bölüm'de bulunan şartlar dahilinde bilgisayar tarafından okunabilir kaynak kodlarının tamamını dağıtacağınıza dair en az üç yıl geçerli olacak yazılı bir taahhütname ile birlikte dağıtmak.

Size verilmiş olan ilgili kaynak kodunu dağıtma taahhütnamesi ile birlikte dağıtmak. (Bu alternatif yalnızca ticari olmayan dağıtımlar için ve yalnızca siz de yazılımı ara derlenmiş veya uygulama biçeminde ve yukarıda b) bölümünde anlatılan şekli ile bir taahhütname ile birlikte almış iseniz geçerlidir.)

Bir eserin kaynak kodu, esere değiştirme yapmak için en uygun yöntem ve imkan anlamında kullanılmaktadır. Uygulama biçeminde bir eser için, kaynak kodu, içerdiği bütün parçalar için ilgili kaynak kodları, ilgili arayüz tanım dosyaları ve derleme ve yükleme işlemlerinde kullanılan bütün betikler anlamında kullanılmaktadır. Bir istisna olarak, dağıtılan kaynak kodu, genelde uygulamanın üzerinde çalışacağı işletim sisteminin ana parçaları (derleyici, çekirdek v.b.) ile birlikte dağıtılan herhangi bir bileşeni, eğer ilgili bileşen, uygulama ile birlikte dağıtılmıyorsa, içermek zorunda değildir.

Eğer uygulama veya ara derlenmiş biçemde yazılımın dağıtımı belli bir yere erişim ve oradan kopyalama imkanı olarak yapılıyorsa, aynı yerden, aynı koşullar altında kaynak koduna erişim imkanı sağlamak, üçüncü şahısların ara derlenmiş ve uygulama biçemleri ile birlikte kaynak kodunu kopyalama zorunlulukları olmasa bile kaynak kodunu dağıtmak olarak kabul edilmektedir.

4. Yazılım'ı bu Lisans'ta sarih olarak belirtilen şartlar haricinde kopyalayamaz, değiştiremez, ruhsat hakkını veremez ve dağıtamazsınız. Buna aykırı herhangi bir kopyalama, değiştirme, ruhsat hakkı verme veya dağıtımda bulunma hükümsüzdür ve böyle bir teşebbüs halinde bu Lisans altındaki bütün haklarınız iptal edilir. Sizden, bu Lisans kapsamında kopya veya hak almış olan üçüncü şahıslar, Lisans şartlarına uygunluklarını devam ettirdikleri sürece, ruhsat haklarını muhafaza edeceklerdir.

5. Bu Lisans sizin tarafınızdan imzalanmadığı için bu Lisans'ı kabul etmek zorunda değilsiniz. Fakat, size Yazılım'ı veya onu baz alan ürünleri değiştirmek veya dağıtmak için izin veren başka bir belge yoktur. Eğer bu Lisans'ı kabul etmiyorsanız bu eylemler kanun tarafından sizin için yasaklanmıştır. Dolayısıyla, Yazılım'ı (veya onu baz alan bir ürünü) değiştirmeniz veya dağıtmanız bu Lisans'ı ve Lisans'ın Yazılım'ı veya ondan iştikak etmiş bütün eserleri kopyalamak, değiştirmek ve dağıtmak için getirdiği şart ve kayıtları kabul ettiğiniz manasına gelmektedir.

6. Yazılım'ı (veya onu baz alan herhangi bir ürünü) yeniden dağıttığınız her defada alıcı, ilk ruhsat sahibinden otomatik olarak Yazılım'ı bu şartlar ve kayıtlar dahilinde kopyalamak, değiştirmek ve dağıtmak için ruhsat almaktadır. Alıcının burada verilen hakları kullanmasına ek bir takım kısıtlamalar getiremezsiniz. Üçüncü şahısları bu Lisans mucibince hareket etmeğe mecbur etmek sizin sorumluluk ve yükümlülüğünüz altında değildir.

7. Eğer bir mahkeme kararı veya patent ihlal iddiası veya herhangi başka bir (patent meseleleri ile sınırlı olmayan) sebep sonucunda size, bu Lisans'ın şart ve kayıtlarına aykırı olan bir takım (mahkeme kararı, özel anlaşma veya başka bir şekilde) kısıtlamalar getirilirse, bu sizi bu Lisans şart ve

kayıtlarına uyma mecburiyetinden serbest bırakmaz. Eğer aynı anda hem bu Lisans'ın şartlarını yerine getiren hem de diğer kısıtlamalara uygun olan bir şekilde Yazılım'ı dağıtamıyorsanız, o zaman Yazılım'ı dağıtamazsınız. Örneğin, eğer bir patent lisansı direkt veya endirekt olarak sizden kopya alacak olan üçüncü şahısların bedel ödemeksizin Yazılım'ı dağıtmalarına hak tanımıyorsa o zaman sizin hem bu koşulu hem Lisans koşullarını yerine getirmenizin tek yolu Yazılım'ı dağıtmamak olacaktır.

Eğer bu bölümün herhangi bir parçası herhangi bir şart altında uygulanamaz veya hatalı bulunur ise o şartlar dahilinde bölümün geri kalan kısmı, bütün diğer şartlar altında da bölümün tamamı geçerlidir.

Bu bölümün amacı sizin patent haklarını, herhangi bir mülkiyet hakkını ihlal etmenize yol açmak veya bu hakların geçerliliğine muhalefet etmenizi sağlamak değildir; bu bölümün bütün amacı kamu lisans uygulamaları ile oluşturulan serbest yazılım dağıtım sisteminin bütünlüğünü ve işlerliğini korumaktır. Bu sistemin tutarlı uygulanmasına dayanarak pek çok kişi bu sistemle dağıtılan geniş yelpazedeki yazılımlara katkıda bulunmuştur; yazılımını bu veya başka bir sistemle dağıtmak kararı yazara aittir, herhangi bir kullanıcı bu kararı veremez.

Bu bölüm Lisans'ın geri kalanının doğurduğu sonuçların ne olduğunu açıklığa kavuşturmak amacını gütmektedir.

8. Eğer Yazılım'ın kullanımı ve/veya dağıtımı bazı ülkelerde telif hakkı taşıyan arayüzler veya patentler yüzünden kısıtlanırsa, Yazılım'ı bu Lisans kapsamına ilk koyan telif hakkı sahibi, Yazılım'ın yalnızca bu ülkeler haricinde dağıtılabileceğine dair açık bir coğrafi dağıtım kısıtlaması koyabilir. Böyle bir durumda bu Lisans bu kısıtlamayı sanki Lisans'ın içerisine yazılmış gibi kapsar.

9. Free Software Foundation zaman zaman Genel Kamu Lisansı'nın yeni ve/veya değiştirilmiş biçimlerini yayınlayabilir. Böyle yeni sürümler mana olarak şimdiki haline benzer olacaktır, fakat doğacak yeni problemler veya kaygılara cevap verecek şekilde detayda farklılık arzedebilir.

Her yeni biçime ayırdedici bir sürüm numarası verilmektedir. Eğer Yazılım bir sürüm numarası belirtiyor ve "bu ve bundan sonraki sürümler" altında dağıtılıyorsa, belirtilen sürüm veya Free Software Foundation tarafından yayınlanan herhangi sonraki bir sürümün şart ve kayıtlarına uymakta serbestsiniz. Eğer Yazılım Lisans için bir sürüm numarası belirtmiyor ise, Free Software Foundation tarafından yayınlanmış olan herhangi bir sürümün şart ve kayıtlarına uymakta serbestsiniz.

10. Eğer bu Yazılım'ın parçalarını dağıtım koşulları farklı olan başka serbest yazılımların içerisinde kullanmak isterseniz, yazara sorarak izin isteyin. Telif hakkı Free Software Foundation'a ait olan yazılımlar için Free Software Foundation'a yazın, bazen istisnalar kabul edilmektedir. Kararımız, serbest yazılımlarımızdan iştikak etmiş yazılımların serbest statülerini korumak ve genel olarak

yazılımların yeniden kullanılabilirliğini ve paylaşımını sağlamak amaçları doğrultusunda şekillenecektir.

GARANTİ YOKTUR

11. BU YAZILIM ÜCRETSİZ OLARAK RUHSATLANDIĞI İÇİN, YAZILIM İÇİN İLGİLİ KANUNLARIN İZİN VERDİĞİ ÖLÇÜDE HERHANGİ BİR GARANTİ VERİLMEMEKTEDİR. AKSİ YAZILI OLARAK BELİRTİLMEDİĞİ MÜDDETÇE TELİF HAKKI SAHİPLERİ VE/VEYA BAŞKA ŞAHISLAR YAZILIMI "OLDUĞU GİBİ", AŞİKAR VEYA ZIMNEN, SATILABİLİRLİĞİ VEYA HERHANGİ BİR AMACA UYGUNLUĞU DA DAHİL OLMAK ÜZERE HİÇBİR GARANTİ VERMEKSİZİN DAĞITMAKTADIRLAR. YAZILIMIN KALİTESİ VEYA PERFORMANSI İLE İLGİLİ TÜM SORUNLAR SİZE AİTTİR. YAZILIMDA HERHANGİ BİR BOZUKLUKTAN DOLAYI DOĞABİLECEK OLAN BÜTÜN SERVİS, TAMİR VEYA DÜZELTME MASRAFLARI SİZE AİTTİR.

12. İLGİLİ KANUNUN İCBAR ETTİĞİ DURUMLAR VEYA YAZILI ANLAŞMA HARİCİNDE HERHANGİ BİR ŞEKİLDE TELİF HAKKI SAHİBİ VEYA YUKARIDA İZİN VERİLDİĞİ ŞEKİLDE YAZILIMI DEĞİŞTİREN VEYA YENİDEN DAĞITAN HERHANGİ BİR KİŞİ, YAZILIMIN KULLANIMI VEYA KULLANILAMAMASI (VEYA VERİ KAYBI OLUŞMASI, VERİNİN YANLIŞ HALE GELMESİ, SİZİN VEYA ÜÇÜNCÜ ŞAHISLARIN ZARARA UĞRAMASI VEYA YAZILIMIN BAŞKA YAZILIMLARLA BERABER ÇALIŞAMAMASI) YÜZÜNDEN OLUŞAN GENEL, ÖZEL, DOĞRUDAN YA DA DOLAYLI HERHANGİ BİR ZARARDAN, BÖYLE BİR TAZMİNAT TALEBİ TELİF HAKKI SAHİBİ VEYA İLGİLİ KİŞİYE BİLDİRİLMİŞ OLSA DAHİ, SORUMLU DEĞİLDİR.

Ek 2 Ses Kartı Sürücüsü Kaynak Kodu

```
/* Aygıt sürücüsü için gerekli include dosyaları */

#include <linux/init.h>

#include <linux/config.h>

#include <linux/module.h>

#include <linux/kernel.h>       /* printk() */

#include <linux/slab.h>         /* kmalloc() */

#include <linux/fs.h>           /* dosya sistemi*/

#include <linux/errno.h>        /* error kodları */

#include <linux/types.h>        /* size_t */

#include <linux/proc_fs.h>

#include <linux/fcntl.h>        /* O_ACCMODE */

#include <linux/ioport.h>

#include <asm/system.h>         /* cli(), *_flags */

#include <asm/uaccess.h>        /* copy_from/to_user */

#include <asm/io.h>             /* inb(), outb() */

#include <linux/interrupt.h>    /* IRQ */


#define ISA_PORT 0x210

#define MODUL_ADI "med"

#define SES_KARTI_INTERRUPT 7


static int interruptsayaci = 0;
```

```
MODULE_LICENSE("Dual BSD/GPL");

MODULE_AUTHOR("Kemal GÜVENLİ");

MODULE_DESCRIPTION("Ses Kartı Sürücüsü");

MODULE_SUPPORTED_DEVICE("AMD Interwave");




/*  med.c fonksiyonlarının deklerasyonları */

int med_open(struct inode *inode, struct file *filp);

int med_release(struct inode *inode, struct file *filp);

ssize_t med_read(struct file *filp, char *buf, size_t count, loff_t *f_pos);

ssize_t med_write(struct file *filp, char *buf, size_t count, loff_t *f_pos);

void med_exit(void);

int med_init(void);



/* Yapı */

struct file_operations med_fops = {

    read: med_read,

    write: med_write,

    open: med_open,

    release: med_release

};
```

```
/* init ve exit fonksiyonlarının deklerasyonu */

module_init(med_init);

module_exit(med_exit);


/* Sürücünün global değişkenleri */

/* Major numarası */

int med_major = 67;

/* Buffer, veri depolamak için */

char *med_buffer;

/*ISAPnP portu*/

int port;


/*-------------------------------------------------------------------------*/


static int interrupt_handler(void)

{

    /* interrupt sayacı */

    interruptsayaci++;

    printk("->->-> SES KARTI KESMESI YONETILDI: interruptsayacı = %d\n", interruptsayaci);
```

```
    /*
     *   "/dev/dsp" tarafından oluşturulan kesmenin icra bölgesi
      */



    return IRQ_HANDLED;
}
/*---------------------------------------------------------------------*/



int med_init(void) {
    int result;
    int ret;
    /* Aygıtı kaydetme */
    result = register_chrdev(med_major, "med", &med_fops);
    if (result < 0) {
            printk("<1>med: major numarası elde edilemiyor %d\n", med_major);
            return result;
    }
    /* Interrupt(Kesme) */
   ret = request_irq(SES_KARTI_INTERRUPT, &interrupt_handler,SA_INTERRUPT, "med",
NULL);
    enable_irq(7);
```

```
printk ("major = %d\n", result);

/* buffer için hafıza(memory) tahsis etme */

med_buffer = kmalloc(1, GFP_KERNEL);

if (!med_buffer) {

        result = -ENOMEM;

        goto fail;

}

memset(med_buffer, 1, 9);



/* BUS için port kontorlu*/

port = check_region(ISA_PORT,1);

printk("port bolge kontrolu = %d\n", port);

if (port){

        printk("<1> med: rezerv edilemez 0x210\n");

        result = port;

        goto fail;

}

request_region(ISA_PORT, 1, "med" );

printk("<1> med modül yerleştirdi\n");
```

```
    return 0;


fail:
    med_exit();
    return result;
}

/*-------------------------------------------------------------------------*/




void med_exit(void) {
    /* major numarası serbest bırakılıyor */
    unregister_chrdev(med_major, "med");


    /*  buffer memory serbest bırakılıyor*/
    if (med_buffer) {
            kfree(med_buffer);
    }
    /* port serbest bırakılıyor*/
    if(!port)
            release_region(ISA_PORT,1);
```

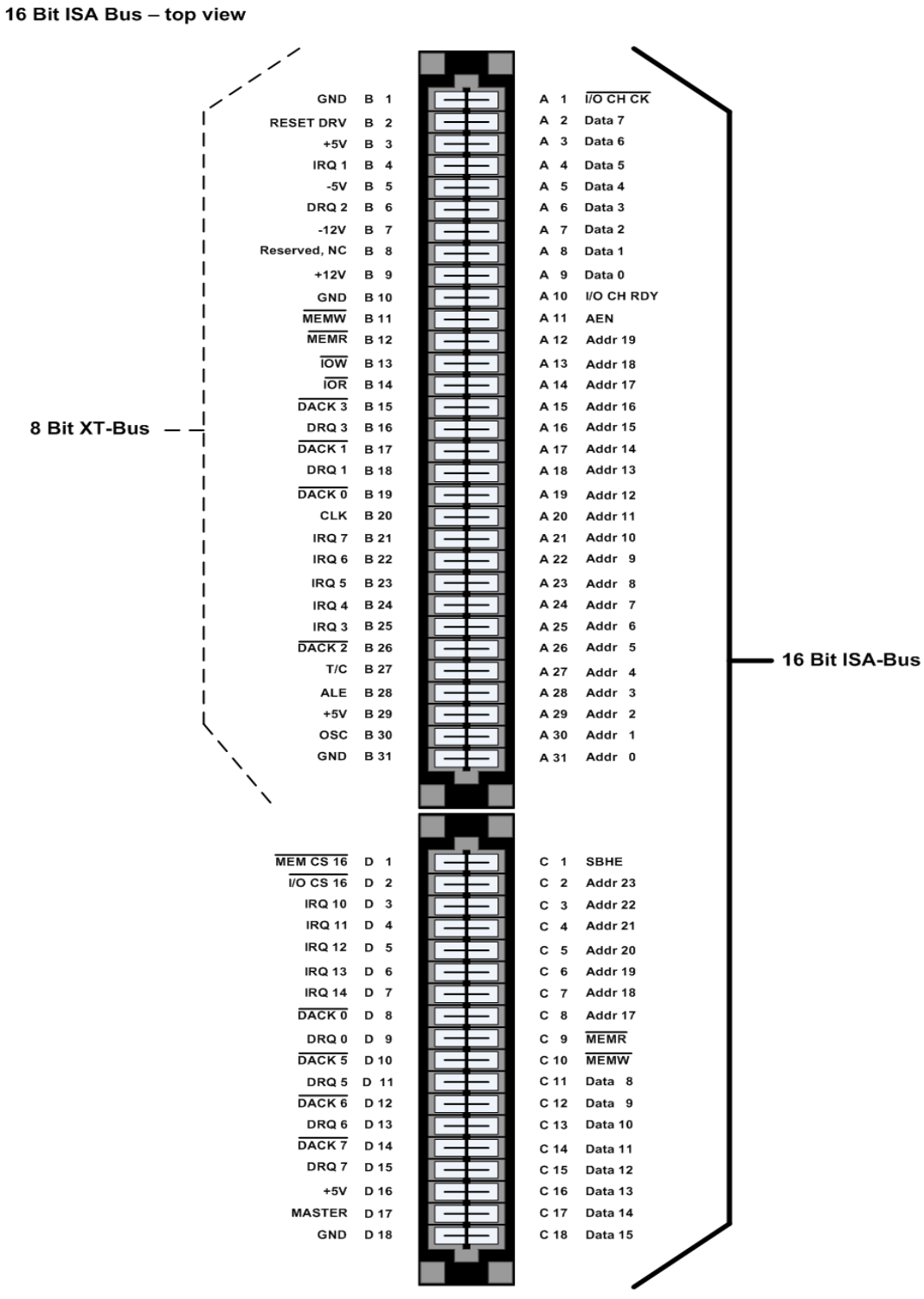
```
    disable_irq(7);
    free_irq(7,NULL);
    printk(med_buffer);
    printk("<1> med modül kaldırılıyor\n");

}

/*-------------------------------------------------------------------------*/

int med_open(struct inode *inode, struct file *filp) {

    /* Başarılı */
    return 0;
}

/*-------------------------------------------------------------------------*/
```

```
int med_release(struct inode *inode, struct file *filp) {

    /* Başarılı */
    return 0;
}

/*-------------------------------------------------------------------------*/

ssize_t med_read(struct file *filp, char *buf,  size_t count, loff_t *f_pos) {

    char med_buffer;
    long int buffer;
    /* data user space e transfer ediliyor */
    copy_to_user(buf,&med_buffer,1);
    /* Port okunuyor*/
    buffer = inb(ISA_PORT);
    printk("Port'tan okunan buffer = %8d\n", buffer);


    /*Okuma pozisyonu en uygun duruma değiştirme */
    if (*f_pos == 0) {
            *f_pos+=1;
            return 1;
```

```
    } else {
            return 0;
    }
}

/*--------------------------------------------------------------------*/

ssize_t med_write( struct file *filp, char *buf, size_t count, loff_t *f_pos) {

    char *tmp;
    char med_buffer;
    tmp = buf + count - 1;
    copy_from_user(&med_buffer, tmp,1);

    /*ISA Port'una  veri yazılıyor*/
    outb(ISA_PORT, 0x03);
    outb(ISA_PORT, 1);
```

```
    printk ("veri porta yazıldı\n");

    return 1;

}



/*-----------------------------------------------------------------*/
```

Ek. 3 ISA BUS



Pin	Name	Direction	Description
A1	/I/O CH CK	Card-to-PC	I/O channel check; active low=parity error

A2	D7	Bidirectional	Data bit 7
A3	D6	Bidirectional	Data bit 6
A4	D5	Bidirectional	Data bit 5
A5	D4	Bidirectional	Data bit 4
A6	D3	Bidirectional	Data bit 3
A7	D2	Bidirectional	Data bit 2
A8	D1	Bidirectional	Data bit 1
A9	D0	Bidirectional	Data bit 0
A10	I/O CH RDY	Card-to-PC	I/O Channel ready, pulled low to lengthen memory cycles
A11	AEN	PC-to-Card	Address enable; active high when DMA controls bus
A12	A19	PC-to-Card	Address bit 19
A13	A18	PC-to-Card	Address bit 18
A14	A17	PC-to-Card	Address bit 17
A15	A16	PC-to-Card	Address bit 16
A16	A15	PC-to-Card	Address bit 15
A17	A14	PC-to-Card	Address bit 14
A18	A13	PC-to-Card	Address bit 13
A19	A12	PC-to-Card	Address bit 12
A20	A11	PC-to-Card	Address bit 11
A21	A10	PC-to-Card	Address bit 10
A22	A9	PC-to-Card	Address bit 9
A23	A8	PC-to-Card	Address bit 8
A24	A7	PC-to-Card	Address bit 7
A25	A6	PC-to-Card	Address bit 6
A26	A5	PC-to-Card	Address bit 5
A27	A4	PC-to-Card	Address bit 4
A28	A3	PC-to-Card	Address bit 3
A29	A2	PC-to-Card	Address bit 2
A30	A1	PC-to-Card	Address bit 1
A31	A0	PC-to-Card	Address bit 0
B1	GND	N/A	Ground
B2	RESET	PC-to-Card	Active high to reset or initialize system logic
B3	+5V		+5 VDC
B4	IRQ2	Card-to-PC	Interrupt Request 2
B5	-5VDC	PC-to-Card	-5 VDC
B6	DRQ2	Card-to-PC	DMA Request 2
B7	-12VDC	PC-to-Card	-12 VDC
B8	/NOWS	Card-to-PC	No WaitState
B9	+12VDC	PC-to-Card	+12 VDC
B10	GND	N/A	Ground
B11	/SMEMW	PC-to-Card	System Memory Write

B12	/SMEMR	PC-to-Card	System Memory Read
B13	/IOW	PC-to-Card	I/O Write
B14	/IOR	PC-to-Card	I/O Read
B15	/DACK3	PC-to-Card	DMA Acknowledge 3
B16	DRQ3	Card-to-PC	DMA Request 3
B17	/DACK1	PC-to-Card	DMA Acknowledge 1
B18	DRQ1	Card-to-PC	DMA Request 1
B19	/REFRESH	Bidirectional	Refresh
B20	CLOCK	PC-to-Card	System Clock (67 ns, 8-8.33 MHz, 50% duty cycle)
B21	IRQ7	Card-to-PC	Interrupt Request 7
B22	IRQ6	Card-to-PC	Interrupt Request 6
B23	IRQ5	Card-to-PC	Interrupt Request 5
B24	IRQ4	Card-to-PC	Interrupt Request 4
B25	IRQ3	Card-to-PC	Interrupt Request 3
B26	/DACK2	PC-to-Card	DMA Acknowledge 2
B27	T/C	PC-to-Card	Terminal count; pulses high when DMA term. count reached
B28	ALE	PC-to-Card	Address Latch Enable
B29	+5V	PC-to-Card	+5 VDC
B30	OSC	PC-to-Card	High-speed Clock (70 ns, 14.31818 MHz, 50% duty cycle)
B31	GND	N/A	Ground
C1	SBHE	Bidirectional	System bus high enable (data available on SD8-15)
C2	LA23	Bidirectional	Address bit 23
C3	LA22	Bidirectional	Address bit 22
C4	LA21	Bidirectional	Address bit 21
C5	LA20	Bidirectional	Address bit 20
C6	LA18	Bidirectional	Address bit 19
C7	LA17	Bidirectional	Address bit 18
C8	LA16	Bidirectional	Address bit 17
C9	/MEMR	Bidirectional	Memory Read (Active on all memory read cycles)
C10	/MEMW	Bidirectional	Memory Write (Active on all memory write cycles)
C11	SD08	Bidirectional	Data bit 8
C12	SD09	Bidirectional	Data bit 9
C13	SD10	Bidirectional	Data bit 10
C14	SD11	Bidirectional	Data bit 11
C15	SD12	Bidirectional	Data bit 12
C16	SD13	Bidirectional	Data bit 13
C17	SD14	Bidirectional	Data bit 14
C18	SD15	Bidirectional	Data bit 15
D1	/MEMCS16	Card-to-PC	Memory 16-bit chip select (1 wait, 16-bit memory cycle)

D2	/IOCS16	Card-to-PC	I/O 16-bit chip select (1 wait, 16-bit I/O cycle)
D3	IRQ10	Card-to-PC	Interrupt Request 10
D4	IRQ11	Card-to-PC	Interrupt Request 11
D5	IRQ12	Card-to-PC	Interrupt Request 12
D6	IRQ15	Card-to-PC	Interrupt Request 15
D7	IRQ14	Card-to-PC	Interrupt Request 14
D8	/DACK0	PC-to-Card	DMA Acknowledge 0
D9	DRQ0	Card-to-PC	DMA Request 0
D10	/DACK5	PC-to-Card	DMA Acknowledge 5
D11	DRQ5	Card-to-PC	DMA Request 5
D12	/DACK6	PC-to-Card	DMA Acknowledge 6
D13	DRQ6	Card-to-PC	DMA Request 6
D14	/DACK7	PC-to-Card	DMA Acknowledge 7
D15	DRQ7	Card-to-PC	DMA Request 7
D16	+5 V	PC-to-Card	
D17	/MASTER	Card-to-PC	Used with DRQ to gain control of system
D18	GND	N/A	Ground