İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

GÖMÜLÜ İŞLETİM SİSTEMLERİ VE μCLINUX'UN AĞ UYGULAMALARINDA KULLANIMI

YÜKSEK LİSANS TEZİ

Bilgisayar Müh. Onur EKİNCİ

Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Programı : BİLGİSAYAR MÜHENDİSLİĞİ

MAYIS 2002

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

GÖMÜLÜ İŞLETİM SİSTEMLERİ VE μCLINUX'UN AĞ UYGULAMALARINDA KULLANIMI

YÜKSEK LİSANS TEZİ
Bilgisayar Müh. Onur EKİNCİ
(504991150)

Tezin Enstitüye Verildiği Tarih : 13 Mayıs 2002
Tezin Savunulduğu Tarih : 31 Mayıs 2002

Tez Danışmanı : Prof.Dr. Bülent ÖRENCİK
Diğer Jüri Üyeleri Prof.Dr. Eşref ADALI (İ.T.Ü)
Yrd.Doc.Dr. Ali Gökhan YAVUZ (Y.T.Ü)

MAYIS 2002

ÖNSÖZ

Tezimin hazırlanmasında verdiği destekten ötürü tez danışmanım Prof. Dr. Bülent Örencik'e teşekkür ediyorum. Öğrencilik hayatımın belki de son ödevini yapmak bana hüzünlü bir mutluluk ve gurur verdi. Bir yarış kazanmış bir maratoncunun mutluluğunu taşımaktayım. Tez çalışmalarımda bana yardımcı olan, aileme, ablam Günseli Ekinci'ye, kız arkadaşım Esra Gemici'ye, arkadaşlarım Muhammet Beratoğlu, Oben Dağ ve Ozan Baltacı'ya, Adam Elektronik A.Ş'ye, benim için düşüncelerinde yer ayıran ve zamanını paylaşan herkese teşekkür ediyorum.

Mayıs 2002 Onur Ekinci

İÇİNDEKİLER

KISALTMALAR

API	: Application Programming Interface
IP	: Internet Protocol
RTLinux	: Real Time Linux
ABS	: Anti Block System
ROM	: Read Only Memory
RAM	: Random Access Memory
DOS	: Disk Operating System
TCP/IP	: Transmission Control Protocol/Internet Protocol
VPN	: Virtual Private Network
Ltd	: Limited
FIFO	: First In First Out
ISR	: Interrupt Service Routine
POSIX	: Portable Operating System Interface
HTTP	: Hypertext Transfer Protocol
SSI	: Secure Socket Interface
HTML	: Hypertext Markup Language
IDE	: Interface Development Environment
CPU	: Central Processor Unit
DLL	: Dynamic Link Library
RAS	: Remote Access System
I/O	: Input/Output
DRAM	: Dynamic Random Access Memory
MMU	: Memory Management Unit
BDM	: Background Debug Module
MSDOS	: Microsoft Disk Operating System
NFS	: Network File System
RISC	: Reduced Instruction Set Computers
MIPS	: Mega Instruction Per Second
FEC	: Fast Ethernet Controller
PC	: Program Counter
SP	: Stack Pointer
CS	: Chip Select
UART	: Universal Asenchron Receiver Transmitter
PPP	: Point to Point Protocol
PC	: Personal Computer
SLIP	: Serial Line Protocol
CGI	: Common Gateway Interface
DNS	: Domain Name System
ICMP	: Internet Control Message Protocol
IST	: Interrupt Service Thread

TABLO LİSTESİ

ŞEKİL LİSTESİ

Sayfa No

SEMBOL LİSTESİ

µClinux	**:** Mikrodenetleyici Linux
µs	**:** Mikrosaniye
b/s	**:** bit/saniye
$	**:** Amerikan para birimi
K	**:** Kilosekizli
bps	**:** bit per second

GÖMÜLÜ İŞLETİM SİSTEMLERİ ve μCLINUX'UN AĞ UYGULAMALARINDA KULLANIMI

ÖZET

Bu çalışmada, gömülü işletim sistemleri incelenmiş ve karşılaştırılmıştır. Gerçek zaman özelliği üzerinde durulmuş, gerçek zaman isterleri belirtilmiştir. Gerçek zamanlı sistemler açıklanmış ve farkları belirtilmiştir. Linux işletim sisteminin gerçek zamanlı bir işletim sistemi olmadığı ve nedenleri belirtilmiştir. Linux'un gerçek zaman dezavantajları için bir çözüm sunan RTLinux ve genel çalışma yapısı açıklanmıştır. Projede kullanılan μClinux işletim sistemi, mimarisi, üzerinde uygulama geliştirme ve derleme konularında detaylı olarak incelenmiştir. Linux ve μClinux işletim sistemleri arasındaki farklar anlatılmıştır. Gömülü işletim sistemi olarak μClinux'un seçilme nedenleri belirtilmiştir. Bir gömülü işletim sistemi olan ve Linux'tan uyarlanan μClinux üzerinde ağ uygulamaları geliştirilmiştir. Motorola 5272C3 geliştirme kiti üzerine μClinux kurulmuştur. Oluşturulan sistemin IP-paylaştırıcı, IP-güvenlik duvarı ve IP-yönlendirici görevi görmesi sağlanmıştır. Bu sistemin yönetimini web üzerinden yapabilmek amaçlı CGI tabanlı programlar yazılmıştır. Sistemin kendisine yüklenilen işlevleri sağlayıp sağlamadığına dair bir test ortamı kurulmuştur. Bu testler sonucunda bu işlevleri sağladığı görülmüştür. Sistemdeki güvenlik duvarı kurallarının her birinin sisteme getirdiği yük incelenmiştir. Gömülü işletim sistemleri ile ilgili incelenen kaynak raporlara ve μClinux kullanarak proje geliştirme tecrübesine dayanarak bir sonuca varılmıştır.

EMBEDDED OPERATING SYSTEMS AND USAGE OF μCLINUX ON NETWORK APPLICATIONS

SUMMARY

In this study, Embedded operating systems are examined and compared. The real time feature and the real time requirements are explained. The real time systems are explained and the differences are clarified. It is expressed that the Linux operating system is not a real time operating system and the reasons are explained. It is examined that one of the solution of the real time disadvantages of Linux is RTLinux and its general architecture. μClinux used in the project is examined regarding its architecture, application development and compiling in detail. The differences between Linux and μClinux are explained. It is pointed why μClinux is choosen as a embedded operating system for the project and the reasons are explained. Network applications are developed on μClinux which is embedded operating system and an adaptation of linux on the embedded environment. μClinux is set up on Motorola 5272C3 development kit. IP-sharer, IP-Firewall and IP-Router functionalities are provided on the installed system. The CGI-based programs are written to manage the system over web. A test environment is set up whether the functionalites are working or not. It is observed that the they are working successfully. It is examined what the implication of one firewall rule on the system load is. Made a decision relying on the examined reports about the embedded operating systems and the experience of project development on μClinux.

1. GİRİŞ

Tezde öncelikle gömülü sistem kavramı incelenmiş, gömülü sistemlerin özellikleri açıklanmıştır. Bu özellikler çerçevesinde gömülü işletim sistemleri "gerçek zaman" özelliğinin sağlanması gereken koşullar çerçevesinde açıklanmış ve karşılaştırmaları yapılmıştır. Ancak uygulamada diğer gömülü işletim sistemlerini kullanmak mümkün olmadığından dolayı bu konuda yapılan çalışmalardan faydalanılmıştır.

Motorola Coldfire gömülü işlemci ailesi üzerinde çalışan linux yani gömülü linux işletim sisteminin temel mimarisi, geliştirilen gömülü sistem üzerinde uygulanması ve bu sistem üzerinde uygulama geliştirilmesi üzerinde durulmuştur.

Gömülü linux olan μClinux'a temel teşkil eden linux işletim sistemi açıklanmış, gömülü bir işletim sistemi olup olamayacağı anlatılmış, "gerçek zaman" özelliği konusundaki eksikleri belirtilmiş ve bu konuda üretilen çözümlerden biri olan RTLinux'tan ve genel yapısından bahsedilmiştir.

Gömülü linux işletim sistemi olan μClinux, detaylı olarak incelenmiş ve bu inceleme sırasında linux işletim sisteminden farklılık gösteren yönleri belirtilmiştir.

Proje geliştirirken gömülü sistem için neden uClinux'ın işletim sistemi olarak seçildiği açıklanmış ve bu platform üzerinde uygulamalar geliştirilmiştir.

μClinux koşan geliştirtiğimiz gömülü sistem üzerinde yapılan uygulamalar anlatılmıştır. Bu uygulamalar, sistemin farklı fonksiyon özellikleri taşıyan ağ cihazları gibi davranmasını ve bu cihazların web üzerinden yönetilmesini sağlamaktadır. Bu farklı fonksiyonlar ise IP-güvenlik duvarı, IP-paylaştırıcı, tam olarak uygulanamamış IP-yönlendiricidir. Sistemin üzerinde koştuğu Motorola 5272C3 geliştirme kiti açıklanmıştır. Yazılan uygulamaların yapısal açıklaması yapılmış ve yazılan programların görevleri belirtilmiştir.

Son olarakta kazanılan tecrübeye ve gömülü işletim sistemleri hakkındaki araştırma raporlarına dayanarak gömülü işletim sistemi olarak yapılan seçim sunulmuştur. Gömülü işletim sistemi olarak μClinux'un doğru seçim olacağı belirtilmiştir.

2. GÖMÜLÜ İŞLETİM SİSTEMLERİNİN GELİŞİMİ

İkinci Dünya Savaşı sırasında ve sonrasında teknoloji çok hızlı gelişmiş ve mühendisler daha önce uğraştıklarından daha karmaşık sistemler tasarlamak ve inşa etmek zorunda kalmışlardır. İnşa edilmek istenilen sistemler bir insanın ve elektromekanik cihazların kontrol edebileceğinden daha karmaşık ve zaman-kritik olduğundan bilgisayarlar bu görevi üstlenmiştir. Bilgisayarlar, sadece esnek ve sınırsız güçlü gözükmekle kalmayıp, bilgisayarların çalışma hızlarına diğer araçlarda ulaşamamıştır. Bunların yanısıra bilgisayarlar, daha ucuz çözümler sunmuştur. Bilgisayarlar bizi elektromekanik cihazların fiziksel sınırlandırmalarından kurtarmıştır. Bilgisayar sistemleriyle donatılmış, amaca özel sistemler, cihazlar ortaya çıkmıştır. Bunlara gömülü sistem, gömülü cihaz denilmiştir [1].

Gömülü sistem, yazılım ve donanımın, gerekirse mekanik ve diğer parçaların oluşturduğu belli bir görevi yapmak üzere geliştirilen sistemdir [2]. Bazı durumlarda gömülü sistemler büyük sistemlerin parçası olabilir. Örnek olarak ABS fren sistemi verilebilmektedir. Bu sistem, fren sistemi ve onu kontrol edecek gömülü bilgisayar sisteminden oluşmaktadır**.** Bu tanımdan ve örnekten yola çıkarak gömülü sistemlere, gömülü bilgisayar sistemleri denilebilmektedir. Gömülü bilgisayar sistemleri, genel amaçlı bilgisayar sistemlerinden farklı olarak belirli bir işi yada işleri yapmak için özelleşmişlerdir. Bu amaçla bu fonksiyonları yerine getirecek indirgenmiş ya da yeniden yazılmış programlar ile donatılmışlardır. Bu programlar FLASH, ROM, RAM gibi bellek birimlerine yazılmaktadır. Gömülü bilgisayar sistemleri için daha belirgin ve daraltılmış bir tanım vermek gerekirse, donanım cihazlarını kontrol etmek amaçlı mikrodenetleyiciler üzerine oturtulmuş belirli fonksiyonları yapan, sisteme özel programlardır [3]. Bu cihazlara örnek olarak otomobiller, ev ve ofis aletleri, cep telefonları verilebilmektedir. Genel amaçlı bilgisayar sistemleri ise, belirli işlemleri gerçekleştirmek için tasarlanmamış, bütün işlemlere zemin hazırlayacak ve bu işlemlerin ortak özelliklerini içeren hem donamımsal hem de yazılımsal bir platform oluştaran sistemlerdir. Gömülü sistemlere örnek olarak, otomotiv sektöründe, 1968 yılında Wolkwagen firması 1600 serisi otomobillerinde benzin enjeksiyon sistemi için mikroişlemciler kullanmışlardır [3]. Haberleşme alanında, 1960 'ların sonlarında elektromekanik telefon anahtarlarının kontrolünde kullanılmış ve bu sistemlere "Yerleştirilmiş Program Kontrol" denilmiştir. Yerleştirilmiş program ile bellek

kastedilmiştir. Bütün program ve yönlendirme bilgileri bellekte tutulmuştur. Bu mantık ile programlar donanıma monte edilmek yerine bellek alanları denilen bölgelere yerleştirilmiştir [4].

Gömülü sistemlerin özellikleri şu şekilde sıralanmaktadır [5]:

i- Üzerinde bulunduğu ortama özgü belirli, tek amaçlı arayüzler içermektedir. Kesinlik, güvenilirlik, kararlılık, band genişliği gibi yüksek teknolojik taleplere yoğunlaşmış sistemlerdir.

ii- Gömülü sistemler, bilgisayar elemanlarının sınırlı kullanılması dolayısıyla sınırlı imkanlara ve fonksiyonlara sahiptir. Bellek kullanımı, enerji tüketimi, kaplanılan alan(küçük elemanların kullanılması) bu sınırlandırmalara örnek teşkil etmektedir.

iii- Gömülü sistemler, hangi amaca yönelik yapılmışsa o amaçla ilgili uygulamayı içermektedir. Projede yapılan sistemde IP-paylaştırıcı işlevselliği istenmektedir. Dolayısıyla sistemde bu işi yapan özel yazılım koşmaktadır.

Birçok gömülü sistem bir işletim sistemine sahip değildir. Sadece bir kontrol çevriminden oluşmaktadır. Bu basit sistemler için yeterli olmakla birlikte sistemin karmaşıklığı ve işlevselliği arttıkça bir işletim sistemine ihtiyaç duyulmaktadır.

Gömülü sistemlerde koşan yazılımlar önceleri sadece bir kontrol çevriminden ibaret olmasıyla beraber değişen ve gelişen teknolojinin bir sonucu olarak bazı gereksinmeler ortaya çıkmıştır. Bunlardan biri de gömülü sistemlerin birçoğunun iletişim için ağ desteğine ihtiyacı olmasıdır. Sadece bir kontrol çevriminden oluşan gömülü sistemlerde ağ katmanını bu çevrime ilave etmek yazılımın karmaşıklığını artırmakta ve uygulanabilirliğini güçleştirmektedir. Dolayısıyla bir işletim sistemine ihtiyaç duyulmuştur.

Bir gömülü işletim sistemine veya gerçek zamanlı bir gömülü işletim sisteminin firmalar tarafından ortaya çıkarılmasındaki nedenleri şu şekilde sıralanabilmektedir:

i- Gerçek zaman gereklerinden zaman kavramını klasik işletim sistemleri yeterli bir şekilde kotaramamaktadır.

ii- Bugünkü ihtiyaçlara göre uygulamaları bir işletim sistemi olmadan yazmak çok zordur ve ihtiyaçları karşılayamamaktadır.

iii- DOS üzerinde uzun zamandır çalışılmamış, ”kendine özgü ” bir çekirdeğe sahiptir. Teknoloji gelişimine çok ihtiyacı olan bir işletim sistemidir.

iv- Windows, güvenilir olmayan, gereksiz ayrıntıları olan, pahalı bir işletim sistemidir.

Gömülü sistemler için işletim sistemleri 1970'li yıllarda ortaya çıkmaya başlamıştır. Bunların çoğu makine dilinde yazılmış ve sadece hangi mikroişlemci için yazılmışsa o mikroişlemcili sistemlerde kullanılabilmekteydi. Dolayısıyla yeni bir mikroişlemci çıktığı zaman yeniden kod yazımı gerektiriyordu. C programlama dilinin ortaya çıkması ile birlikte, işletim sistemleri daha etkin, kararlı ve taşınabilir amaçlı yazılmaya başlanmıştır. C programlama dili işletim sistemlerinin yazılması konusunda standartlaşmıştır. Bu şekilde yazılan kodların tekrar kullanılabilirliği artmıştır.

Ticari gömülü işletim sistemleri 1980 yıllarında ortaya çıkmış ve günümüze kadar gelişmiştir. Bu işletim sistemlerine örnek olarak Vxworks, Qnx, Windows CE, μClinux verilebilmektedir [4].

Linux işletim sistemi, Linus Torvalds tarafından eğitim amaçlı yazılmış Minix işletim sisteminden esinlenerek ortaya çıkarılmıştır [6]. Linux'un bellek yönetim birimi olmayan mikroişlemcili gömülü sistemlere taşınmış versiyonu μCLinux'tur. Linux işletim sisteminin bellek yönetim sistemi olmayan işlemciler üzerine uygulama çalışmaları 1997 yılında başlamıştır. İlk çalışmalar Motorola 68000 ailesi işlemciler üzerine yapılmış ve ilk adaptasyon SCADA denetleyiciler üzerine 1997/98 yıllarında yapılmıştır. Yine 1998 yılında, μClinux'un Palm Pilot için alternatif bir işletim sistemi olarak bir sürümü çıkmıştır. Bu tarihten sonra μClinux birçok ticari uygulamada kullanılmıştır. Gömülü linux, gömülü cihazlara ileri düzeyde ağ yeteğini standart olarak sunmuş ve sistem tasarımcılarına çok büyük miktarlarda ücretsiz ulaşılabilecek yazılımlar sunmuştur. İletişimin ön planda olduğu günümüz internet dünyasında, gömülü linux kararlı TCP/IP yığın kümesi ve uygulama katmanında çalışan çok geniş spectrumlu ağ servisleri ile ön plana çıkmıştır. Bu doğrultuda, gömülü linux'u işletim sistemi olarak kullanan Motorola 68000 tabanlı ağ yönlendiricileri, VPN geçit kapıları, web kameraları, IP üzerinden ses aktarımı ve çok çeşitli endüstriyel kontrol uygulamalarında kullanılmıştır [7].

Bir gömülü sistem, gerçek zamanlı olabilmekte veya olmayabilmektedir. Bu sistemin üzerinde koşan yazılımlara yani işletim sistemi ve uygulama katmanı yazılımına, bununla birlikte donanıma bağlıdır. Sistemdeki donanım, gerçek zamanlı bir sistem yaratmakta bize kolaylık sağlayabilmekte veya zorluk çıkarabilmektedir. Aynı donanım üzerinde yazılıma bağlı olarak gerçek zamanlı ya da gerçek zamanlı olmayan sistem yaratmak mümkündür.

Gerçek zamanlı sistem, dışarıdan gelen girişlere, etkilere karşı önceden belirlenebilen ve tahmin edilebilen sürelerde cevap verebilen sistemlerdir [8].

Gerçek zamanlı sistemlerden beklenen özellikler [8]:

i- Kesin zaman çizgilerini karşılayabilmek:

Bir olay olduğu anda sistemin önceden belirlenmiş belirli bir zaman sınırında tepki vermesi gerekmektedir. Bu sınır değerini aşması halinde sistem geri dönüşümü olmayabilmekte ve sistem, işlevselliğini kaybedebilmektedir.

ii- Aynılık ya da aynı anda işleme:

Birden fazla olay aynı anda olursa bütün olaylar için zaman sınırlandırmalarına uymak şartı ile olaylara gerekli tepkileri vermektir. Bundan dolayı gerçek zamanlı sistemlerin doğal paralelizme ihtiyacı vardır. Bu da ya çok işlemci kullanmak ya da çok-işlem yaklaşımını adapte etmeyi gerektirmektedir.

Gerçek zamanlı sistemleri, kesin, esnek gerçek zamanlı sistemler olarak ikiye ayrılmaktadır [8].

Kesin gerçek zamanlı sistemlerin özellikleri:

i- Sistemdeki herhangi bir gecikme kabul edilmemektedir.

ii- Gecikme sonucunda gereksiz ve faydasız sonuçlar elde edilmektedir.

iii- Zaman sınırlamasının aşılması halinde sistemin çökmekte ve sistemi yeniden ayağa kaldırmak imkansız hale gelmektedir(Donanımsal tekrar başlatım hariç).

iv- Zaman sınırlamasının aşılmasının yarattığı maliyet çok yüksek olmaktadır.

Esnek gerçek zamanlı sistemlerin özellikleri:

i- Sistemdeki gecikme maliyeti yükseltmektedir.

ii- Gecikme halinde sistemin yeniden ayağa kaldırılabilmekte ve sistem kabul edilebilir zaman-tolerans aralıklarına sahip olabilmektedir. Örnek olarak, ses uygulamalarında paketlerin arada sırada kaçması sesin kalitesini çok fazla etkilememektedir. Ancak bu durumun sık olması durumunda etkilemektedir. Ağ arayüzü alt sistemi örnek olarak verilebilir. Paketler kaçırıldığı zaman, paketlerin yollanmasını tekrar isteyerek tekrar kazanım yapılabilmektedir. Fakat bu şekilde sistemin performansı düşmektedir. Görüntü ve ses uygulamalarındaki bu performans düşmesi kaliteyi etkiler fakat sistemin çökmesine neden olmaz.

Kesin ve esnek gerçek zaman sistem farkı ise, üzerinde çalışılan sistemin isteklerine bağlıdır. Eğer sistem belirli bir zaman sınırını aşmamak zorunda ise “kesin”, zaman sınırını aşmamalı ise “esnek” gerçek zamanlı sistemdir.

3. GÖMÜLÜ İŞLETİM SİSTEMLERİ

Gömülü gerçek zamanlı cihazlar gün geçtikçe artmaktadır. Bu cihazlar, bilgisayar sistemlerine göre daha hızlı, daha akıllı ve daha ucuz satılmaktadır. TCP/IP yığını ve kablosuz ağ cihazlarına ihtiyaç artmaktadır. Bu nedenlerden dolayı, gömülü işletim sistemlerinde gerekler, ihtiyaçlar artmaktadır, işletim sistemlerinin desteklediği özellikler artmaktadır. Protokol popülaritesi, işletim sistemlerinin tasarımında önemli bir faktör olarak karşımıza çıkmaktadır. Müşteriler, istedikleri protokollerin, kullandıkları gömülü sistemlerde desteklenmesini istemektedirler. TCP/IP, 802.11 bu protokollere örnek olarak verilebilir.

Donanım gün geçtikçe gelişmekte, fiyatlar düşmektedir. Bu durum da gerçek zamanlı işletim sistemleri üreticilerinin teknolojiyi takip etmesini güçleştirmektedir. Cihaz sürücü geliştirimi ve işletim sistemlerinin yenilenmesini, yeni versiyonlar çıkarılmasını güçleştirmektedir. Sadece iyi desteklenmiş gerçek zamanlı işletim sistemleri ayakta kalmaktadır. Kendi işletim sistemlerini geliştiren firmalar zorluk çekmeye başlamakta ve geniş geliştirme kaynakları olmayan firmalar, "kendine özgü" çözümleri geliştirmeleri imkansız hale gelmektedir. Örnek olarak Motorola, μClinux işletim sistemi üzerinde çalışan gruplara destek vermektedir. Bu gruplarda Motorola'nın mikroişlemci satışlarını arttırmak için bu işlemcilere özgü μClinux yazmaktadırlar.

Bütün gerçek zamanlı işletim sistemlerinin amacı daha gelişmiş özellikleri barındırmaktır. Fakat bunu sahip oldukları güvenilirlik ve sağlamlık özelliğini bozmadan yapmaya çalışmaktadırlar.

Bir işletim sisteminin gerçek zamanlı olabilmesi için bazı istekleri yerine getirmesi gerekmektedir. Bunlar [8]:

i- İşletim sistemi çok görevcikli(çok görevli de denilebilir. Görevler süreçlerden oluşmaktadır. Çoğu işletim sistemi de cok görevcikli süreçlerden oluşmaktadır. Sadece aynı bellek uzayını kullanan görevciklerden oluşan işletim sistemleride vardır) ve elinden alınabilir olmalıdır:

Gerçek zamanlı bir işletim sistemi kestirilebilir olmalıdır. Bu işletim sisteminin çok hızlı olmasını gerektirdiği anlamına gelmez, bir işi yaparken ne kadar sürede yapabileceği önceden kestirilebilir olmalı ve bu süre uygulama isteklerini karşılayabilecek düzeyde olmalıdır. Programlayıcı, çalışan görevcikten kaynak

kullanımını alıp diğer bir görevciğe verebilmelidir. İşletim sistemi ve donanım mimarisi birden fazla kesme seviyesine izin vermeli ve elinden alma olayını kesme seviyesinde gerçekleyebilmelidir.

ii- Görevciklerde öncelik kavramı var olmalıdır:

Esas problem, görevciklerden hangisinin kaynağa en çok ihtiyacı olduğunu belirlemektir. Gerçek zamanlı işletim sistemlerinde zaman sınır çizgisine en yakın olan sürece kaynak verilmesi gerekmektedir. Bununla birlikte işletim sistemi bir süreç işini bitirmek zorunda ise bunun için ne kadar zamana ihtiyaç duyulduğunu bilmesi gerekmektedir. Bu da uygulanması zor bir durum teşkil etmektedir. Bu yüzden süreç öncelik seviyesi kavramı geliştirilmiştir. Zaman sınır çizgi gerekleri, süreç öncelik seviyelerine dönüştürülmüştür.

iii- İşletim sistemi, tahmin edilebilen, görevcik uyum mekanizmalarını desteklemelidir:

Görevcikler, aralarında verileri ve kaynakları paylaştıkları zaman birbirleriyle haberleşme ihtiyacı duymaktadırlar. Bundan dolayı görevcikler arası haberleşme mekanizmaları var olması gerekmektedir. Aynı zamanda bu mekanizmalarının davranışları kestirilebilir olmalıdır. Örneğin eski UNIX sistemlerinde sadece süreçler vardır. Bu süreçler farklı bellek alanlarında çalıştıkları için birbirleriyle borular ve paylaşılan bellek yöntemleri ile haberleşmektedirler. Bu iki mekanizmada dosya sistemini kullandıklarından tahmin edilemeyen davranışlar ortaya çıkmaktadır.

iv- Öncelik değerini alma mekanizmasının var olması gereklidir:

Öncelik dönmesi olayını önleyebilmek için gerekli bir mekanizmadır [9].

v- Zamanlama ihtiyaçlarının göz önünde tutulması gerekmektedir:

Sistem çağrılarının yürütülme zamanı ve sistemin davranışının bilinmesi gerekmektedir. Bundan dolayı kesme gecikmeleri, sistem çağrılarının aldığı zaman gerçek zaman işletim sistemi üreticileri tarafından sağlanması gerekmektedir.

Qnx 6.1, Vxworks 5.3.1 ve Windows CE 3.0 işletim sistemlerinin, iş yönetimi, bellek yönetimi, kesme kotarımı, sağladığı API'lar, araçlar, tablolar halinde karşılaştırılmalı olarak Ek A'da bulunmaktadır [10].

3.1 Qnx 6.1

QNX Yazılım Sistemleri Ltd, 1980 yılında Gordon Bell ve Dan Dodge tarafından, gerçek zamanlı bir işletim sistemi geliştirmek, sürdürmek ve pazarlamak amaçlı kurulmuştur.

QNX gerçek zamanlı işletim sisteminin sunduğu, gerçek zamanlılık, yüksek güvenilirlilik, yüksek peformans, istenilen işlevsellikleri karşılama ve ileriye yönelik kolay geliştirilebilme özelliklerinden dolayı 3Com, Motorola, Cisco gibi firmalar tarafından, tüketici elektroniği, haberleşme, otomotiv sistemleri gibi konularda tercih edilmiştir.

3.1.1 Kurulum ve Ayarlar

QNX 6.1, çabuk ve kolay bir kuruluma sahiptir. İşletim sistemini, QNX yazılım firmasından satın alınacak yoğun diski de internet sitesinden indirmek mümkündür [11]. Temel modüller, çekirdek ve kullanıcı arabirimi birkaç dakika içinde kurulabilmektedir. Derleyici gibi diğer kısımlar sağlanılan paket yöneticileri tarafından kurulabilmektedir.

Tüm parçalar kurulduktan sonra, sistemin önemli kısımlarını oluşturan depolama cihazları, ağ kartı gibi donanımlar otomatik olarak taranmaktadır. Yapılması gereken başka ayarlar varsa sağlanılan grafik arayüzü kullanılarak yapılabilmektedir. İşletim sisteminin görüntüsü, belirli konfigürasyon dosyaları değiştirilerek yapılıyor. Eklenmesi, değiştirilmesi, çıkarılması gereken modüller bu dosyaların içeriklerini değiştirerek yapılıyor. Firmanın sağladığı dokümanlarda bu dosyalara ilişkin birkaç yetersiz örnek bulunmaktadır. Bu ayarlar için bir grafik arayüzü sunulmamıştır.

3.1.2 Mimari

İstekli/Sunucu mimarisine sahiptir. Bir mikroçekirdekten ve diğer opsiyonel olarak seçilebilen süreçlerden oluşmaktadır. Mikroçekirdek sadece temel servisleri yerine getirmektedir. Bunlardan bazıları sinyaller, mesaj aktarımı, eşzamanlama, çizelgeleme, zamanlayıcı servisleridir. Mikroçekirdek kodu, sadece bir çekirdek çağrısı, donanım kesmesi yada işlemci ikazı sonucu yürütülmektedir.

Diğer süreçler, sunucu süreçler olarak görev yaparlar ve istekli süreçlere(uygulama süreçleri gibi) cevap verirler. Bu tür süreçlere örnek olarak dosya sistem yöneticisi, süreç yöneticisi, cihaz yöneticisi ve ağ yöneticisi gösterilebilmektedir. Intel işlemcilerde, çekirdek öncelik seviyesi 0'da, yöneticiler ve cihaz sürücüler seviye 1 ve 2'de, diğer uygulama süreçleri seviye 3'te çalışmaktadır. Bu yüzden uygulama süreçleri sadece genel komutlara erişebilmektedirler.

QNX 6.1, mesaj tabanlı bir işletim sistemidir. Mesaj aktarımı, süreçler arası haberleşmede temel araç görevi görmektedir. Mesaj aktarım servisi, istekli/sunucu modeline dayanmaktadır. İstekli(örneğin uygulama süreci) sunucuya(örneğin cihaz yöneticisi) bir mesaj gönderir. O da uygun bir cevap yollar. QNX API çağrıları, mesaj aktarım mekanizmasını kullanır. Örneğin, bir uygulama süreci bir dosya

açmak istediği zaman, sistem çağrısı bir mesaja dönüştürülerek dosya yöneticisine yollanır. Dosya yöneticisi(cihaz sürücüsü üzerinden diske ulaştıktan sonra), gerekli dosyayla cevap verir. QNX 6.1, mesaj aktarımı sırasında, "istekçi tarafından sürülen öncelik" mekanizmasını kullanır. Bu mekanizmada, sunucu süreç, kendisinden servis isteyen istekli sürecin öncelik seviyesini alır. Servis verildikten sonra tekrar geri kendi öncelik seviyesini kazanır. Eğer birden fazla istekli süreç, aynı sunucudan servis isterse, hizmetçi en yüksek önceliğe sahip istekli sürecin öncelik seviyesini alır. Bu da öncelik dönmesi probleminden kaçınmayı sağlamaktadır.

İstekli/sunucu modelinin birçok avantajı vardır. Bunlardan biri sistemin dayanıklı, kuvvetli olmasıdır. Her yönetici(süreç yöneticisi hariç) ve cihaz sürücüleri kendi sanal bellek adres uzaylarında çalışmaktadır. Bu da kuvvetli ve güvenilir bir sistem sağlamaktadır.

Sistem çağrılarının yürütülmesi bağlam anahtarlamalarına yol açar. Bellek koruma sisteminden dolayı sisteme yük binmektedir. Dolayısıyla sistemin performansı etkilenir.

QNX 6.1, süreçleri ve görevcikleri kullanmaktadır. Süreçler, görevciklerin çalışacağı adres uzayını tanımlar ve en az görevcik içerir. Uygulamalar için 63 farklı görevcik öncelik seviyesi vardır. Görevcikler, öncelikleştirilmiş FIFO ve round robin yöntemine göre çizelgeleştirilmişlerdir. İş yönetimi ile ilgili bilgiler Tablo A.1'de gösterilmiştir.

Her süreç kendi sanal bellek alanında çalışmaktadır. Sanal bellek, süreçleri birbirine karşı koruyarak sistemin sağlamlığını arttırmaktadır. Bellek yönetimi ile ilgili bilgiler Tablo A.2'de bulunmaktadır.

Mikroçekirdek içindeki kesme yönlendiricisi, kesme kotarımı için görevlendirilmiştir. Yönlendirici kesme geldiğinde o anda çalışan görevciğin bağlamını saklar. Daha sonra ISR'yi içeren görevciğin bağlamı, yönlendirici tarafından işlemci içeriğine yüklenir.

QNX 6.1, mesaj tabanlı bir işletim sistemi olduğundan, ISR ile görevciklerin haberleşmesi sinyal ve darbeler ile olmaktadır. ISR, süreçlere ve görevciklere sinyal yollayabilir. Tutturulduğu sürece, uygulamaya özgü veri aktarımı yapabilir. Tablo A.3'te kesme kotarımı ile ilgili bilgiler verilmektedir.

3.1.3 API Zenginliği

Qnx 6.1, POSIX uyumlu ve kendine özgü API sunmaktadır. Tablo A.4a'da iş yönetimine ilişkin, Tablo A.4b'de sabit blok uzunluklu, Tablo A.4c'de değişken

uzunluklu bellek parçaları için bellek yönetimine ilişkin çeşitli API desteği bilgileri verilmiştir.

3.1.4 Internet Desteği

QNX 6.1'in internet teknoloji katmanı şu ürünleri ve araçları içermektedir:

Voyager web sunucusu: HTTP sunucusudur. SSI üzerinden dinamik web sayfalarını desteklemektedir.

Bilgi görmek için Voyager web tarayıcısı: HTML 3.2, çerçeve yapısı, javascript desteği vardır. Mozilla ve Opera Internet tarayıcıları, QNX 6.1 üzerinde çalışabilmek için uyarlanmışlardır.

Voyager yazılım geliştirme seti: Gömülü sistem üzerinde, bilgisayar ağları ve internet üzerinden haberleşebilen uygulamalar geliştirmekte kullanılmaktadır.

3.1.5 Araçlar

QNX 6.1 için iki tane araç seti bulunmaktadır. Bunlar:

i- MetroWorks CodeWarrior IDE

ii- GCC araç kiti.

Her iki kit içinde farklı ortamlarda geliştirme imkanı vardır. Bu araç kitleri çokça kullanılan araçları içermektedir. Bu araçlar Tablo A.5'te gösterilmektedir. Bu araç setlerine ek olarak TCP/IP geliştirme kiti de mevcuttur. Bu kit vasıtasıyla TCP/IP bağlantısı üzerinden haberleşen uygulamalar geliştirilebilmektedir.

3.1.6 Dokümantasyon

Sisteme genel bakış ve sistemin mimarisi konusunda dokümanlar iyi hazırlanmıştır. Ancak API ortamı çok iyi açıklanmamıştır. API parametrelerinin anlamlarının bir kısmı açıklanmamıştır. Dokümantasyonda bir kaç çelişkili ifade mevcut ve dokümanların tekrar gözden geçirilmesi gerekmektedir.

QNX 6.1 işletim sistemi, teknik destek konusunda yeterli gözükmektedir. Arttırılmış destek seçeneği için ek ücret talep edilmektedir.

3.1.7 Lisans ve Maliyet

QNX 6.1, ticari olmayan kullanımlar için kendi web adresinden indirilebir [11]. Ticari kullanımlardaki maliyet için şirketle direkt temasa geçilmelidir.

3.1.8 Sonuç

QNX 6.1, istekçi/sunucu mimarisine sahip bir işletim sistemidir. Her süreç, cihaz sürücüleri de dahil olmak üzere, kendi sanal bellek alanlarında çalışmaktadır. Sistem birçok noktaya dağıtılabilmekte ve ağ-şeffaf bir yapıya sahip olduğundan dağıtılmış sistemler için çok iyi bir mimari teşkil etmektedir(mesaj aktarım).

Sistem performansı hızlı ve kestirilebilir.

Daha önceki versiyonları sadece x86 ailesi işlemcilerini desteklemekle beraber QNX 6.x, MIPS, PowerPC, SH4, StrongARM ‘yi de desteklemektedir.

Dokümantasyon geliştirilmesi gerekmekte ve API’lar için verilen bilgiler daha açık ve detaylı olması gerekmektedir.

3.2 Vxworks 5.3.1

Vxworks 5.3.1, Wind River Sistemleri firması tarafından çıkarılmış, yüksek performanslı, çok geniş spektrumlu donanım platformlarında çalışan karmaşık gerçek zaman ve gömülü uygulamaları için bir yürütme ortamı teşkil eden gerçek zamanlı bir işletim sistemidir [12].

3.2.1 Kurulum ve Ayarlar

Tornado 1.0.1 ve VxWorks 5.3.1, tek yoğun disk içinde gelmektedir. VxWorks, hedef cihaz üzerinde çalışan gerçek zamanlı bir işletim sistemidir. Tornado ise bilgisayar üzerinde çalışan geliştirme ortamıdır.

Kurulumun ilk adımı olarak kullanıcıdan şifre istenmektedir. Bu şifreye göre hangi parçaların kurulumuna izin verildiği belirlenmektedir. Kullanıcı ürün listesinden hangi ürünleri kuracağını seçebilmektedir.

Kurulumun ikinci adımı olarak lisans yöneticisi kurulması gerekmektedir. Her lisanslanmış ürün için bir şifre gerekmektedir. İki çeşit lisans uygulaması vardır. Bunlar, kayan lisans, tek kullanıcı lisansıdır.

Tek kullanıcı lisansı, bir makine üzerindeki bir kullanıcı için geçerlidir. Bu lisans, ağ ortamı ve kayan lisans sunucusu olmadığı zaman ve sadece bir makina üzerinde kullanılmaktadır. Kayan lisans, ağ ortamına sahip, birden fazla kullanıcı için kullanılır. Tek lisans sunucu servisi, istekli uygulamalarının lisansların geçerliliğini sınamaktadır. Lisanslar, satın alınan miktara göre kullanıcılara dağıtılmıştır.

Ayarlar, kart destek paketleri aracılıyla yapılır. Bu paketler, belirli platformlar için ayar dosyaları içermektedir. Kullanıcı, kaynak/hedef ortamının kurulumu için büyük esnekliklere sahiptir. Hedef, ROM'den ya da ağ üzerindeki bir kaynaktan başlatılmaktadır.

3.2.2 Mimari

Wind River felsefesi, birbirini tamamlayıcı ve işbirliği içinde iki işletim sistemi kullanmaktır. Bunlardan biri, üzerinde Windows NT ya da UNIX gibi klasik işletim sistemleri koşan kaynak makinadır. Bunun üzerinde geliştirme ortamı kurulmuştur. Diğeri ise üzerinde Vxworks işletim sistemi koşan ve gerçek zaman işlerini yerine getiren hedef makinadır. Sadece hata keşfi servisi hedef platformda çalışmaktadır. Diğer bütün geliştirme ortamı kaynak makinadadır. Başlangıçta Vxworks, pSOS işletim sistemi için geliştirme ve ağ ortamı iken, Wind River Sistemleri firması, VxWorks için kendi mikroçekirdeklerini geliştirmişlerdir. Dolayısıyla Vxworks işletim sistemi, ağ özelliğinin başlangıçta var olmasından dolayı istekli/sunucu mimarisine sahiptir.

Sahip olduğu mikroçekirdek, çok işlem yapma, çizelgeleme, karşılıklı iş iletişimi ve eş zamanlaması ve bellek yönetimi gibi özellikleri barındırmaktadır. Diğer bütün işlevler süreç olarak uygulanmaktadır.

Vxworks 5.3.1, çok fazla zahmet harcanmadan geliştirilebilir ve adapte edilebilir bir işletim sistemidir. Modüler yapıdan dolayı, modüller çıkarılarak ya da eklenerek, bellek sınırlaması olan küçük gömülü sistemlerden, daha karmaşık sistemlere göre ayarlar yapılabilmektedir.

Vxworks, farklı işlemciler üzerindeki süreçler arası haberleşme konusunda desteği zayıftır. Örneğin, bu işlemciler için bir paylaşılan bellek varsa iletişim için mesaj kuyrukları kullanılabilmektedir. Paylaşılan bellek objelerini kullanmak için "VxMP" denilen bileşen kurulması gerekmektedir. Paylaşılan bellek kullanılmadan soket tabanlı iletişimde kullanılabilir. Fakat bu yöntem mesaj kuyruklarına nazaran bir çok dezavantaja sahiptir. Bunlardan biri de sağlanılan veri yapısı yetersizliğidir. Buna ek olarak TCP/IP'nin güvenli iletişim ortamlarında mesaj yollanırken getirdiği yüktür.

Vxworks 5.3.1 gerçek zaman çekirdeği, temel, çok işlem ortamı sunmaktadır. Her işin kendi bağlamı vardır ve işler çekirdek tarafından çizelgelendirilmektedir. İş yönetimi ile ilgili bilgiler Tablo A.1'de gösterilmiştir.

Vxworks, POSIX ve kendine özgü iş çizelgeleme algoritmalarını kullanır. Bunların yanında farklı algoritmalarıda kullanır. Bunlar:

i- Alma - öncelikli programlama:

Vxwork için öndeğer algoritmasıdır. Bu algoritma, en yüksek önceliğe sahip hazır konumdaki görevciğin CPU'yu almasını sağlar. Eğer bir görevcik o anda çalışan görevcikten daha yüksek önceliğe sahip ise çekirdek önceliği daha yüksek olan görevciğin bağlamına anahtarlanır. Aynı önceliğe sahip süreçler için FIFO çizelgeleme algoritması kullanılır.

ii- Döngü programlama :

Bu algoritma, alma-öncelikli algoritmanın geliştirilmesidir. Bu algoritma, eşit önceliğe sahip görevciklerin CPU'yu paylaşımını daha etkin kılmaya çalışmaktadır. Bunu da eşit seviyeli görevcikler için zaman paylaşımı yaparak sağlamaktadır. Bir görevciğe tanınan zaman aralığı dolduğu zaman, çekirdek, diğer eşit seviyeli görevciği çalıştırmaktadır [13].

Vxworks 5.3.1'de bütün sistem ve uygulama işleri aynı adres uzayını paylaşır. Dolayısıyla kötü yazılmış bir uygulama sistemin kararlılığını bozabilir. Fakat ayrıca sipariş edilen ek bir bileşen olan VxVMI ile bir süreç kendi çalışma uzayına sahip olabilmektedir. Bellek yönetimi ile ilgili bilgiler Tablo A.2'de bulunmaktadır.

Görevcik olan işler aynı adres uzayını kullanırlar. Bir iş, süreç olabilmesi için kendi çalışma uzayı olması gerekir. Dışarıdan gelen kesmelere karşı hızlı cevap verilebilmesi için kesme servis rutinleri, görevciklerin bağlamı dışında bir özel bağlamda çalışır. Bütün kesme rutinleri aynı yığını kullanırlar. Ayarlanabilen parametreler ile sistem başlangıcında bu yığın yerleştirilebilinir. Yığın hatalarını önleyebilmek için, yığın boyutu olabilecek en kötü iç içe kesme kombinasyonlarına göre ayarlanmalıdır. Kesme servis rutinleri aynı yığını paylaştıklarından dolayı bazı önemli sınırlandırmalara sahiptirler. Kesme servis rutinleri, çağıranın bloke olabileceği rutinlere işaret etmemesi gerekir. Örneğin, bir kesme servis rutininin çağırdığı program parçası bir semaforu yakalamak istiyor ve bu semafor kullanımda ise, çekirdek bu rutini bekleme statüsüne almaktadır. Tablo A.3'te kesme kotarımı ile ilgili bilgiler verilmektedir.

3.2.3 API Zenginliği

Çok geniş kapsamlı bir API'ya sahiptir. Tablo A.4a'da iş yönetimine ilişkin, Tablo A.4b'de sabit blok uzunluklu, Tablo A.4c'de değişken uzunluklu bellek parçaları için bellek yönetimine ilişkin çeşitli API desteği bilgileri verilmiştir.

3.2.4 Internet Desteği

Wind River Sistemleri, "gömülü sistemler için Tornado" isimli bir ürüne sahiptir. Bu ürün şu parçaları kapsamaktadır:

i- Bilgileri internet ortamına aktarmak için bir HTTP sunucu.

ii- İnternet üzerindeki bilgileri görebilmek için bir web tarayıcı.

iii- "Bir kere yaz her yerde çalıştır" uygulamaları geliştirmek için java desteği.

iv- İnternet kullanıcılarına bilgileri gösterebilmek için grafiksel araçlar.

v- İnternete hazır, gerçek zamanlı işletim sistemi ve geliştirme kiti.

vi- Özel donanımlar için sürücüler.

vii- İnternet üzerinden haberleşme için ağ protokol desteği.

Bunlara ek olarak, Vxworks işletim sisteminin, C++ kütüphane, HTML tabanlı grafikler ve pJava desteği vardır.

3.2.5 Araçlar

Wind River, gömülü uygulamalar için sisteme bütünleştirilmiş geliştirme ortamı sunmaktadır. Bu sistemin adı da "Tornado"dur. Tornado herkese açık bir ortam olduğundan, herkes tarafından araç geliştirilebilir, ilave edilebilir ya da isteğe göre ayarlanabilir.

Tornado'nun bir çok özelliği kendine özgü araç komut dili tarafından uygulanmıştır. Kullanıcılar bu dili kullanarak kendine özgü araçlar geliştirebilirler.

3.2.6 Dokümantasyon

Tornado 1.0.1 çok fazla kullanma kılavuzları ile gelmektedir. Ancak bu kılavuzlar yeteri kadar açık ve detaylı degildir. Örneğin API ve kütüphane çağrıları için verilen referans kılavuzda, her fonksiyonun parametrelerinin açıklaması yoktur. Sistem mimarisi ile ilgili çok az bilgi verilmiştir. Kullanıcı aramakta olduğu bilgiye ulaşabilmek için birçok kılavuzu karıştırmak zorunda kalabilmektedir.

Wind River'ın web sayfası çok geniş kapsamlı ve güncel bilgi deposu sunmaktadır [14]. Bu bilgi deposuna ulaşabilmek için kullanıcı adı ve şifre almak zorunluluğu vardır.

Müşteri desteği çok profesyonelce yapılmakta, özellikle elektronik posta ile yapılan destek çok iyidir. Tornado destek isteği mesajları Tornado çalışma ortamından da yollanabilmektedir. Tornado destek isteği mesajları, Wind River teknik destek ekibine yollanmadan önce, Tornado destek araçları tarafından kaynak ve hedef sisteminize özel bilgiler, teknik destek ekibine yardımcı olması amacıyla ilave edilmektedir.

3.2.7 Lisans ve Maliyet

Lisans, çalışan sistem başına verilmektedir.

Sistemin çalıştığı her platform başına ücret alınmaktadır. Sayı arttıkça fiyat düşmektedir.

3.2.8 Sonuç

i- VxWorks istekli/sunucu mimarisine sahiptir. Mikroçekirdek, temel gerçek zaman ihtiyaçlarını yerine getirmektedir(kesme kotarımı, iş yönetimi gibi...). Diğer işlevler süreçler olarak uygulanmaktadır. Vxworks, Qnx gibi mesaj tabanlı bir işletim sistemi değildir.

ii- Vxworks 5.3.1 hızlı ve tahmin edilebilen davranışlara sahip bir işletim sistemidir.

iii- Çok geniş kapsamlı bir API ve kütüphanelere sahiptir. Yine çok geniş spektrumlu cihazlar için cihaz sürücülerine sahiptir.

iv- Tornado 1.0.1 iyi ve çok fazla araç sunan bir geliştirme ortamıdır. Ayrıca açık kaynak olmasından dolayı, dışarıdan ilaveler yapılabilmektedir.

v- Wind River teknik destek ekibi çok profesyonel ve internet üzerinden günün her saati ulaşabilinecek çok geniş kapsamlı ve iyi düzenlenmiş bilgi deposu sunmaktadır.

vi- Kullanma kılavuzlarındaki bilgiler çok açık yazılmamıştır. Sistem mimarisi ve bazı API fonksiyonları detaylı açıklanmamıştır.

vii- Bütün işler herhangi bir bellek koruma mekanizması olmaksızın aynı adres uzayını paylaşmaktadırlar. Bellek koruma, ek olarak sipariş edilecek olan VxVMI bileşeni ile sağlanabilmektedir.

viii- MIPS, x86, Motorola gibi çok çeşitli donanım platformu için uygun bir işletim sistemidir.

3.3 Windows CE 3.0

Microsoft firması tarafından üretilmiş bir gömülü işletim sistemidir [15].

3.3.1 Kurulum ve Ayarlar

Windows CE 3.0 işletim sistemini kullanabilmek için ilk adım, platform kurucu yazılımını kurmaktır. Platform kurucusu, seçime özgü Windows CE platformu yaratmak için birçok araçtan oluşan bir settir. 14 yoğun diskten oluşmakta ve ARM, MIPS, PowerPC, SH ve Intel x86 tabanlı platformlar oluşturabilmektedir. Platform kurucusunun diğer Windows uygulamaları gibi kolay ve kullanıcı dostu bir kurma süreci vardır.

İkinci adım ise, bu platform kurucusunu kullanarak, istenilen platformun kurulması ve ayarlarının yapılmasıdır. İstenilen özelliklere göre oluşturulan platformun ayarlarının yapılması karmaşık bir süreçtir. Birçok kurma sihirbazı içermesine rağmen, çoğu sistem ayarı kullanıcı tarafından yapılması gerekmektedir. Bunlar, sistem kütüğü dosyaları, ortam değişkenleri ayarları ve bazı ayar senaryolarıdır. Bu durum yeni başlayanlar için ayar sürecini zorlaştırmaktadır.

İstenilene özel platform inşa etmeyi basitleştirmek için platform kurucusu sekiz farklı Windows CE işletim sistemi ayarlarını içermektedir. Başlangıç noktası olarak bu ayarlardan başlanılabilmektedir.

3.3.2 Mimari

Windows CE ileriye yönelik ve ihtiyaçlara göre geliştirilebilir bir işletim sistemidir. Bir dizi farklı modüllerden oluşmaktadır. Bunların her biri farklı fonksiyonlar görmektedir. Bu modüller parçalara ayrılmakta ve parçalar ayrı ayrı seçilmektedir. CE 3.0, en kısa boyut için 200 kilosekizlik bir ROM alanı istemektedir.

Dört ana modül önemli özellikleri sağlamaktadır:

i- Çekirdek, işletim sisteminin en iç kısmını oluşturmaktadır. Bellek yönetimi, görevcik çizelgeleme, süreç yönetimi, ikaz değerlendirme ve kesmelerden sorumludur. Parçalara bölünebilen DLL modülünden oluşmaktadır.

ii- Nesne deposu, Windows CE tarafından sağlanılan üç sürekli alanı belirtmektedir. Bunlar sistem saklayıcısı, dosya sistemi ve Windows CE özellik veri tabanıdır. Windows CE veritabanı, yapılandırılmış yerleşim sağlamakta ve bu verilere giriş, verilerin sıralandırılması sağlanmıştır. Bu verilere Windows CE ‘nin veri tabanı API’ları ile de ulaşılabilmektedir.

iii- Grafikler, pencereleme ve olaylar alt sistemi, işletim sistemi ile kullanıcı, uygulama arasında bir arayüz teşkil etmektedir.

iv- İletişim kısımları ise TCP/IP, RAS, Seri I/O gibi protokollere destek vermektedir.

Windows CE, çok süreçli ve çok görevcikli bir işletim sistemidir.

Her zaman, bir süreç yaratıldığında bir de görevcik yaratılmaktadır. Bir süreç içindeki görevcikler, o sürecin kaynaklarını ve adres uzayını paylaşmaktadırlar.

Her görevciğin kendi yığını ve öncelik seviyesi vardır. Çekirdek, bu öncelik seviyelerini kullanarak, görevcikleri döngü yöntemine göre çizelgelemektedir. Eşit seviyeli görevcikler için zaman dilimleri kullanılmaktadır. Bu zaman dilimleri ayarlanabilmektedir. Eğer sıfır değerine ayarlanırsa görevcikler arasında yarış başlamaktadır.

Bütün işletim sistemi görevcikleri, çekirdek ya da kullanıcı modunda çalışabilmektedir. Çekirdek modunda çalıştırma sistem performansını arttırmakla beraber sistemin kırılganlığınıda arttırmaktadır. İş yönetimi ile ilgili bilgiler Tablo A.1'de gösterilmiştir.

Her süreç kendi sanal bellek alanına sahiptir. Windows CE adres alanını 32 megasekizlik parçalara bölmektedir. Bu alanların her biri bir süreç için ayrılmaktadır. Her süreç sadece 32 megasekizlik sanal adres uzayına sahiptir. Bu uzay, kümeler, yığınlar ve sanal bellek yerleştirimi için kullanılır. Eğer 32 megasekizliden daha fazla alana ihtiyaç olursa haritalandırılmış bellek dosyaları aracılığı ile boyut değişimi yapılabilmektedir. Bellek yönetimi ile ilgili bilgiler Tablo A.2'de bulunmaktadır.

Windows CE, kesme kotarımının kesme servis rutini içinde değil kesme servis görevciği içinde yapılabilmesi için görevcik kesme modelini kullanmaktadır. Windows CE, bir kesme ile bir olayı ilişkilendirmek için bir API sunmaktadır. Kesme geldiğinde çekirdek, kesme servis rutinini çalıştırır. Bu rutin bittikten sonra kesme ile ilişkilendirilmiş olayı işaret eder. Daha sonra çekirdek bu olayı bekleyen keme servis görevciğini çalışmak üzere çizelgelendirir. Eğer bu sürecin öncelik seviyesi, kesilmiş görevcikten yüksekse hemen çalışmaktadır. OEM, yapılan işi azaltmak için doğrudan kesme servis rutinini kullanabilir. Fakat kesme servis rutinine müdahale edebilecek bir API sağlanmamıştır. Kesme servis rutinine sadece OEM adaptasyon katmanını değiştirerek müdahale edilebilmektedir. Tablo A.3'te kesme kotarımı ile ilgili bilgiler verilmektedir.

3.3.3 API Zenginliği

Windows CE, Windows32 API'sının bir alt kümesini barındırmaktadır. API, çokça kullanılan çekirdek özelliklerini sağlamaktadır. Tablo A.4a'da iş yönetimine ilişkin, Tablo A.4b'de sabit blok uzunluklu, Tablo A.4c'de değişken uzunluklu bellek parçaları için bellek yönetimine ilişkin çeşitli API desteği bilgileri verilmiştir.

3.3.4 Internet Desteği

Windows CE, çok geniş kapsamlı internet desteğine sahip ürünler ve araçlarla gelmektedir. Bunlar:

i- Web sayfalarını aktarabilmek için HTTP sunucu vardır. Bu sunucu aktif sunucu sayfalarını desteklemektedir.

ii- İnternet üzerindeki bilgileri görmek için bir web tarayıcısı vardır. Bu tarayıcı internet inceleyicisinin küçük bir versiyonudur. Bu tarayıcı, çerçeve, tablo formatlarını, javascript, ses dosyalarını desteklemektedir.

iii- Uzaktan cihazları yönetebilmek için telnet sunucuya sahiptir.

iv- İnternet üzerinden haberleşebilmek için ağ protokol desteği vardır.

3.3.5 Araçlar

Tablo A.5'te Windows CE için kullanılan araçlar görülmektedir. Buna ek olarak, platform kurucusu, uzaktan erişim araçlarına sahiptir. Bunlar:

i- Uzak küme yürüyücüsü, cihaz üzerinde çalışan süreçler için küme belirteçlerini ve bayrakları hakkında detaylı bilgi verir. Bu şekilde bir sürecin aldığı bellek alanını verip vermediği görülebilmektedir.

ii- Uzak casus isimli araç, kaynak üzerinde çalışmakta ve cihaz üzerinde çalışan uygulamaların gönderdiği mesajları almaktadır.

iii- Uzak dosya göstericisi ile bilgisayardan, Windows CE 'nin koştuğu cihaz ya da cihazlardaki klasörler hiyerarşik bir şekilde görülmektedir. Dosyalar silinebilmekte ve içerikleri görülebilmektedir. Bu aracın, görüntü ekranı olmayan cihazlarda kullanımı çok faydalıdır.

iv- Uzak saklayıcı editörü ile Windows CE'nin koştuğu cihazın sistem ayarları görülebilmektedir.

3.3.6 Dokümantasyon

Windows CE, web üzerinden her an ulaşabilecek bir dokümantasyon sağlamaktadır. Çok fazla bilgi içermesine rağmen bu bilgiler iyi yapılandırılmamıştır. Dokümantasyon bir referans olarak kolayca kullanılabilmekte fakat eğitici kaynak olarak fazla uygun değildir. Dolayısıyla, yeni başlayanlar, çok zaman harcamak zorunda kalmaktadır. Sistem içindeki çalışma hakkında derinlemesine ve detaylı bilgi verilmemekte, API'lar ve kullanılabilecek özellikler hakkında anlaşılmasını kolaylaştıracak bilgi yeterliliği verilmemektedir.

Microsoft, "Microsoft Geliştirme Ağı" web sitesinde detaylı bilgi deposu sağlamaktadır. OEM'ler için ek ödeme karşılığında destek verilmektedir.

3.3.7 Lisans ve Maliyet

Bütün özellikleriyle platform kurucusu için belli bir ücret alınmaktadır. Üzerinde Windows CE koşan cihazlar içinde ayrıca çalışma lisansı alınması gerekmektedir.

3.3.8 Sonuç

i- Windows CE birçok özelliğe sahip ve yine bir çok donanım paltformunu destekleyen bir işletim sistemidir.

ii- Windows CE, gerçek zamanlı bir işletim sisteminden beklenen özellikleri göstermektedir.

iii- Karmaşık ve ayarlama özelliği yüksek olan bir işletim sistemidir. Windows CE'nin ayarlama süreci çok karmaşıktır. Çünkü bu konuyla ilgili dokümantasyonda yapılandırılmış bir yaklaşım bulunmamaktadır. Yeni başlayanlar, sistemi tam olarak doğru ayarlamak için kendilerini sistem saklayıcısı ve diğer ayar dosyalarını incelerken bulabilmektedirler.

iv- Dokümantasyon, sistemin iç yapısında yapılan işler hakkında detaylı ve derinlemesine bilgi yetersizliğine sahiptir. API'lar iyi dokümante edilmiştir. Fakat bunların, kullanıcı için anlaşılırlığı yeterince sağlanamamıştır.

v- Sistem, sağlam bir yapıya sahiptir. Yüksek sistem yoğunluğunda hatasız çalışabilmektedir. Windows CE kendisini hatalı uygulamalara karşı korumak için sanal bellek korunumunu kullanmaktadır.

4. GÖMÜLÜ LINUX

4.1 Linux

Linux işletim sistemi, ilk olarak Linus Torvalds tarafından, Helsinki Üniversitesi'nde ortaya çıkarılmıştır[6]. Linux, aslında bir işletim sistemi çekirdeğidir. Seçilen kütüphaneler ve uygulamalar ile bir sistem oluşturmaktadır. Genel olarak bu sistem "Linux sistem" olarak adlandırılır. Linux çekirdek kodu ve bir çok uygulama, "Herkese açık genel lisans" adı altında piyasaya çıkmaktadır. Bunun anlamı, isteyen bu kodları ücretsiz olarak elde edebilir ve isteklerine göre bu kod üzerinde değişiklik yapabilir demektir.

Linux, gerek performansı, gerek güvenilirliği gerekse sahip olduğu geniş destekle sadece kişisel bilgisayarlarda değil karmaşık fonksiyonları olan güçlü bilgisayarlarda da işletim sistemi olarak kullanılmaya başlanmıştır. Linux'ın sahip olduğu bu özellikler ile gömülü sistemler geliştirenlerinde dikkatini çekmiştir. Linux'un gömülü bir işletim sistemi olarak kullanılabilecek bir işletim sistemi olup olmadığı düşünülmüştür.

Linux temel özellikler ve ağ desteği ile 500 kilosekizli RAM boyutu işgal etmektedir [16]. Gömülü sistemlerin bellek sınırlaması olduğundan bu durum linux için bir avantaj teşkil etmektedir. Linux sahip olduğu modüler yapıdan dolayı istenilen çekirdek işlevleri ve sürücüler derleme sırasında ilave edilebilmektedir. Örneğin ağ katmanı, ağ katmanı içinde de yönlendirme özelliği derleme sırasında eklenebilmektedir. Bu modüler yapıda linux'a kolay geliştirilebilirlik ve esneklik kazandırmıştır. Bu durum linux'un gömülü sistemlere adaptasyonunu kolaylaştırmıştır. Linux, birçok gömülü sistem işlevlerini karşılayabilecek bir işletim sistemidir. Örnek olarak ağ desteği ve kullanıcı için bir arayüz sağlaması verilebilmektedir. Linux, sorun çıkarmadan uzun süre çalışabilmektedir. Bu durum da gömülü sistemler için gerekli kararlılık koşulunu sağlamaktadır. Linux, fiyat ve sunduğu özellikler bakımından gömülü sistemler için uygun bir işletim sistemidir. Linux'un gömülü işletim sistemi olarak kullanılmasının birçok avantajı vardır. Bunlar:

i- Linux, açık kaynak bir işletim sistemidir. Bu şekilde işletim sistemine yeni işlevler eklenebilmektedir. Hata ayıklama ve düzeltme daha kolay yapılabilmektedir. Ayrıca işletim sisteminin içinde neler olduğu anlaşılabilmekte ve tek bir işletim sistemi

üreticisine bağlı kalınmamaktadır. Açık kaynak olma özelliği sayesinde yazılımların işbirliği ile yapılarak daha etkin ve iyi olması sağlanabilmektedir.

ii- Linux, birçok donanım platformunda çalışabilmektedir ve çok geniş uygulama yazılımı deposuna sahiptir.

iii- Linux, ücretsizdir. Geliştirilen birçok yazılımda maddi çıkar gözetmeksizin yapılmaktadır.

iv- Linux'u bir platformdan diğerine taşımak kolay ve hızlıdır.

v- Çok geniş spektrumlu cihaz sürücü yazılımına sahiptir.

vi- Unix tabanlı olduğundan ağ desteği çok iyidir.

vii- Modüler ve platformdan bağımsız bir yapıya sahip olduğundan, yapılan işin karmaşıklığı artsa ya da azalsa bile çok fazla efor sarfetmeden yeni iş kotarılabilmektedir.

viii- Linux, iyi dokümante edilmiş kaynak kodlardan dolayı istenilen şekilde özelleştirilebilmektedir. Örneğin üzerinde linux çalışan bir bilgisayar veya kart, IP-paylaştırıcı ya da IP-güvenlik duvarı olarak özelleştirilebilmektedir.

Genel amaçlı işletim sistemleri, ortalama performansı en uygun yapmaya ve her sürece eşit yürütme zamanı ayırmaya çalışırlar. Linux'ta bu mantıkta yazılmış bir işletim sistemidir. Fakat gerçek zamanlı çizelgelemede kesin zamanlama ve kestirilebilir performans, ortalama performansa göre daha önemlidir. Linux'un bu "adil zaman" prensibi gerçek zamanlı işletim sistemi olmasını önleyici nedenlerden biridir. Kritik bir işe en yüksek öncelik bile verilse, linux'un "eşit" zaman paylaşımlı çizelgeleme algoritması nedeniyle bu iş gecikmeli çalışabilir. Bu algoritma, her kullanıcı programının adil bir şekilde kaynakları kullanmasını sağlamaktadır. Ancak istenilen, gerçek zamanlı bir işin adil olsun ya da olmasın, ihtiyacı olduğu zaman CPU'yu kullanmasıdır.

Linux'un genel olarak kullandığı POSIX standartlarının gerçek zaman konusundaki dezavantajları şu şekilde sıralanabilmektedir [17]:

i- Linux süreçleri, bağlam yönünden ağır süreçlerdir. Bu da süreç anahtarlama konusunda önemli bir ağırlık getirmektedir. Linux, diğer klasik işletim sistemlerine göre süreç anahtarlamada hızlı olmasına rağmen, hızlı bir makinada bu işlem yaklaşık birkaç yüz mikrosaniye sürmektedir. Bu durumda, bir sensörü her 200 mikrosaniyede bir taraması gereken bir sürecin çizelgelendirilmesini imkansız hale getirmektedir.

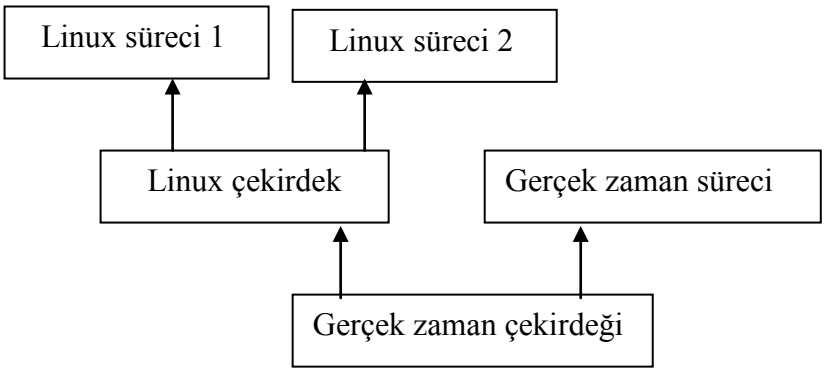
ii- Linux, standart unix tekniğinde olduğu gibi, çekirdek süreçlerini "elinden alınmaz" durumda çalıştırmaktadır. Yani bir süreç, sistem çağrısı yaptığı zaman,

sistem çekirdek modunda çalışmakta, önceliği ne olursa olsun başka bir sürecin kullanımı için işlemci elinden alınamamaktadır. Çekirdek modunda tekrar çizelgelendirme yapılamaz. Örnek olarak bir kullanıcı programı "fork" sistem çağrısı yaptığı zaman, "fork" çağrısı tamamlanana kadar bir başka süreç çalışamaz.

iii- Linux, çekirdek kodunun kritik kısımlarında kesmeleri kapatmaktadır. O anda çalışan süreç, öncelik seviyesi ne olursa olsun, kritik kısmında ise gerçek zaman kesmesi gelse bile kritik kısmını bitirene kadar kesme bekletilmektedir. Linux çekirdeğindeki bu kesme kapamaları ve kritik kısımların varlığından dolayı, en kötü gecikme süresini hesaplamak zordur. İyi bir tahmin için cihaz sürücüsü ve işletim sistemi kodu iyi incelenmelidir.

Bütün bunların ışığında Linux'u, "elinden alma" özelliğine sahip ve kesme işleme süreci gecikmelerinin daha az olduğu bir gerçek zaman çekirdeğine dönüştürmek gerekmektedir. Fakat bu durum da çekirdeğin yeniden yazılmasını gerektirmektedir. Sistemin yeniden yaratılması linux dünyasında yeni sorunlar doğuracağından RTLinux ortaya çıkarılmıştır.

RTLinux'taki temel fikir, Linux'u gerçek zaman çekirdeği kontrolü altında çalıştırmaktır [17]. Şekil 4.1'de temel çalışma şekli görülmektedir.



Şekil 4.1- RTLinux temel çalışma yapısı

RTLinux'ta, yapılması gereken gerçek zamanlı bir iş varsa yapılmaktadır. Eğer yoksa, gerçek zaman çekirdeği Linux'u çalışması için çizelgelemektedir. Linux, gerçek zamanlı çekirdeğin en düşük öncelikli işidir.

Linux'un kesmeleri kapaması problemi ise gerçek zamanlı çekirdek zerinde çözülmüştür. Linux çekirdeği, kesmeleri kapamak için "cli()" rutinini yürüttüğü zaman bir yazılım kesme bayrağı sıfıra çekilmektedir. Bütün kesmeler gerçek zaman çekirdeği tarafından yakalanır, bu bayrağa göre ve kesme maskesine göre kesmeler linux çekirdeğine geçirilmektedir. Bu yüzden bütün kesmeleri Linux kapamaya kalksa bile her zaman açık kalmaktadır. Bir kesme olduğunda gerçek zaman çekirdeği onu yakalamakta ve ne yapılması gerektiğine karar vermektedir. Eğer

kesme, bir gerçek zaman işinin çalışmasına neden oluyorsa, Linux'un bulunduğu durum saklanır ve gerçek zaman işi başlatılmaktadır. Eğer kesme Linux için ise, gerçek zaman çekirdeği bekleyen kesmenin olduğunu belirten bayrağı kurar ve Linux kesme kotarıcısı çalıştırılmadan Linux'un çalışmasına devam edilmektedir. Linux kesmeleri açtığı zaman, gerçek zaman çekirdeği bekleyen kesmeleri işletmekte ve ilişkisel Linux kesme kotarıcısının çalışmasına neden olunmaktadır.

Gerçek zaman çekirdeği de "elinden alınmaz" yapıdadır. Fakat içerdiği rutin küçük ve hızlıdır. Dolayısıyla büyük gecikmelere neden olmamaktadır. Pentium 120 üzerinde yapılan deneylerde en fazla çizelgeleme gecikmesi 20 μs ölçülmüştür.

4.2 Mikrodenetleyici Linux(μCLINUX)

Linux, çekirdeğinin basitliği, çekirdek parçalarının ve uygulamaların rahatlıkla ayarlanabilir olması nedeniyle, kaynak sınırlamaları olan ortamlara uygulanmasını sağlamıştır.

μClinux, gömülü ortamlar için uyarlanmış bir sistemdir. Linux çekirdeği, bellek yönetim birimi olmayan işlemciler için değiştirilmiştir. μClinux, temel olarak Linux 2.0.38 çekirdeğini almıştır. μClinux, 32 bit gömülü işlemcileri hedef almıştır. Daha sonraları başka işlemci ailelerini de desteklemiştir. Bunlar:

i- Gömülü Motorola 68000 işlemciler(68302, 68328, 68332, 68360)

ii- Intel i960

iii- ARM

iv- Motorola Coldfire işlemciler(5206, 5206e, 5307)

μClinux, Linux çekirdeğinden, buna ek olarak kodlardan ve sanal bellek desteği olmadan çalışmasını sağlayan değişikliklerden oluşmaktadır. Temel çekirdek fonksiyonları olan süreç yönetimi, dosya sistemleri, cihaz sürücüleri aynı kalmıştır [7].

Standart sanal bellekli Linux sistemi ile μClinux arasındaki en önemli fark, bellek koruma mekanizmasının olmamasıdır. Herhangi bir süreç, diğer bir sürecin ya da çekirdeğin bellek alanına girebilmektedir. Bu durumu bellek yönetim birimi olmadan çözecek basit bir yol mevcut değildir. Ancak birçok gömülü sistemde, sistem kurucuları hangi yazılımın çalışacağını ve nasıl çalışacağını kontrol etmektedirler .

4.2.1 Sistem

μClinux tabanlı gömülü sistemler 2 temel parçadan oluşmaktadır. Bunlardan birisi işletim sistemini temel fonksiyonlarını içeren ve sistemin tabanını teşkil eden

çekirdektir. Diğeri ise çekirdek üzerinde çalışan uygulamaların depo edildiği kök dosya sistemidir.

Birçok gömülü sistem "hepsi bir yerde" yaklaşımını benimsemiştir. Bu doğrultuda çekirdek ve uygulamalar aynı görüntü içinde birleştirilmiş, işletim sistemi işlemleri ile uygulama seviyesi işlemleri arasında bir ayrım konmamıştır. Ancak μClinux'ta, birçok masaüstü ve sunucu işletim sistemlerindeki gibi çekirdek ve uygulama seviyesi işlemleri arasında kesin bir ayrılık vardır.

μClinux'ta uygulamalar kendi öncelik seviyelerinde, kendi bellek alanlarında çalışmaktadırlar. Çekirdekten servis almak istediklerinde çekirdeğe atlanılmaktadır.

μClinux etrafında kurulan sistem bir kök dosya sisteme ihtiyacı vardır. Normal olarak ROM tabanlı dosya sistemi kullanılmıştır. Bu da ROM ve kalıcı belleğin gömülü sistemlerde birarada bulunmasını kolaylaştırmıştır. Kök dosya sistemi, ağ dosya sistemi veya depolama cihazları olabilmektedir.

4.2.2 Çekirdek

μClinux için, Linux çekirdeğinde yapılmış ana değişiklik farklı bir bellek yönetim alt sistemi kullanılmıştır. Çekirdeğin, sanal bellek eşleştirmeleri, sayfa tabloları ya da yeni bir süreç için yeni sanal bellek alanı yaratma gibi durumlara ihtiyacı kalmamıştır. Ancak boş ve dolu bellek havuzlarını belirlemeye ve takip etmeye ihtiyaç vardır. Çekirdek, yine bellek isteklerine cevap vermek ve kullanılmış bellekleri serbest bırakmaktan sorumludur.

Sanal bellek mekanizmasına sahip olmamanın bazı dezavantajları vardır. Bunlardan en önemlisi ise standart Linux süreç yaratma mekanizması ile ilgilidir. Linux'ta yeni bir süreç yaratma "fork" sistem çağrısı ile yapılmaktadır. Normalde, fork sistem çağrısı, çağıran sürecin bir kopyasını yeni bir sanal bellek alanına çıkarmaktadır. Ancak yeniden yaratma işlemi sanal bellek olmadan net biçimde gerçekleştirilememektedir. Çalışan bir sürecin bellek görüntüsü için kesin işaretçilerin hesaplanması olayında etkili ve pratik bir yol yoktur.

μClinux, "fork" sistem çağrısı yerine "vfork" sistem çağrısını desteklemektedir. "vfork" sistem çağrısı, görevcik için yeni bir sanal bellek alanı yaratmamakta, bunun yerine aynı bellek adres alanını kullanmaktadır. Bunu yapabilmek için görevcik ya yeni bir süreç olana kadar("exec" sistem çağrısı üzerinden) ya da görevcikten çıkılana kadar ana sürecin çalışması askıya alınmaktadır.

Diğer önemli bir dezavantaj ise, büyük bellek isteklerinin küçük bellek parçalarının birleştirilerek karşılanmasının sağlanamamasıdır. Bütün bir bellek alanı isteği karşılanabilmesi için o boyutta boş bellek alanı mevcut olması gerekmektedir.

Bundan dolayı bellek yerleştirimi çok dikkatli yapılmalıdır. Mümkün olduğunca küçük parçalardan kaçınılmalıdır. Ancak μClinux bu konuda zayıftır. Bunun nedeni ise Linux'un standart 2'nin kuvvetlerine göre yerleşim mekanizmasını kullanmasıdır.

Sanal bellek mekanizmasında, her uygulamanın süreç yığını, istenildiği zaman dinamik olarak büyüyebilmektedir. Bu durum bellek yönetim birimi olmadan mümkün degildir. μClinux'ta, her uygulama programı sabit bir yığına sahiptir. Bu yığının boyutu öndeğer olarak 4 kilosekizlidir. Fakat derleme sırasında bu boyut her süreç için değiştirilebilinmektedir.

Diğer ana çekirdek parçaları olan, ağ katman, süreç yönetimi, zaman paylaşımı ve sistem çağrısı işleme standart Linux'taki gibidir.

4.2.3 Sistemi Başlatma

μClinux kullanan sistemlerde birçok başlangıç metodu bulunmaktadır. En basit metodta ise, işlemci başlangıç vektörü ROM veya kalıcı bellekte bulunan μClinux çekirdeğine atlamaktadır. Çekirdek ise işlemcinin ve çevre birimlerin normal donanım ayarlarını(yonga seçimleri, DRAM denetleyici gibi...) kurmakla sorumludur. Kök dosya sistemi çekirdek ikili kodlarından sonra ROM veya kalıcı bellekte yer almaktadır.

Daha detaylı olarak, işlemci başlangıç vektörü vasıtasıyla bir başlangıç yükleyicisi çalıştırılır. Daha sonra çekirdek ve kök dosya sistemi RAM'ye yüklenir. Bu durum sistemin performansını arttırmaktadır. Çünkü çekirdek, ROM veya kalıcı bellekten ziyade RAM'den daha hızlı yürütülmektedir. Bir başka avantaj ise, çekirdek ve kök dosya sistemi sıkıştırılmış bir yapıda ROM veya kalıcı belleğe yerleştirilebilinmekte ve başlangıç yükleyicisi tarafından RAM'ye açılabilmektedir. Bu durum boyut tasarrufunu dolayısıyla maliyet tasarrufunu sağlamaktadır.

μClinux çekirdeği bir cihazı konsol olarak kullanabilmek için ayarlanabilmektedir. Coldfire işlemcilerde seri portlardan biri konsol olarak kullanılabilinmekte ve başlangıç işlemi tamamlandıktan sonra etkileşimli kabuk gelmektedir. Bu durum hata ayıklama ve çözme konusunda kolaylıklar sağlamaktadır. Bununla beraber sistem herhangi bir konsol cihaz olmadan da başlayabilmektedir. Bu, birçok gömülü sistemde mevcut olan tipik bir durumdur.

Motorola 5272C3 geliştirme kiti kullanılarak sistemin başlangıç süresi, başlangıç yükleyicisi ve görüntünün kalıcı bellekten RAM'ye yüklenmesi ve işletim sisteminin başlaması olarak belirlenmektedir. Bu süre yaklaşık 5 saniye ölçülmüştür.

4.2.4 Uygulamalar

µClinux üzerinde birçok uygulamanın derlenmesi ve çalıştırması saydamdır. Sistem, diğer Linux sistemler gibi gözükmektedir. Bu da µClinux'un en önemli karakteristiklerinden biridir. Bu durum µClinux'un gömülü sistemler kurulumunda kullanımını kolaylaştırmıştır. Linux üzerinde çalışan bir uygulama gömülü bir sistem üzerinde de kolaylıkla kullanılmaktadır.

Uygulama seviyesi API'sı, standart Linux sistem çağrıları setinden oluşmaktadır. Bu unix sistemlerde de aynıdır. Ancak bellek yönetim mekanizmasından dolayı, bellek yönetim sistem çağrılarında değişiklik göstermektedir. "fork" sistem çağrısı yerine "vfork" kullanılmaktadır. Ayrıca bellek yerleştirimi ve serbest bırakımı konusunda da "malloc" ve "free" kütüphane rutinleri kullanılmaktadır.

µClinux, bellek yönetim birimi olmayan işlemciler için geliştirilmiştir. Bellek yönetim birimi olmayan işlemciler geliştirilmesine gerek duyulmuştur. Bunun nedeni ise daha ucuz olmaları ve sanal bellek sayfası yer değiştirmelerinden kaynaklanan kesmelerin gerçek zaman uygulamaları için çok zaman kaybettirmesi ve uygun olmamasıdır. Bellek yönetim birimi olmamasından dolayı kod alanları korumasız kalmakta ve izinsiz bellek girişleri olmaktadır. Dolayısıyla Linux'un bellek yönetim birimi değiştirilmiş(Linux'un "/mm" klasörü "/mmnommu" ile değiştirilmiş ve bu klasörde MMU'nun eksikliği giderilmeye çalışılmış ve temel bellek yönetim fonksiyonları sağlanılmaya çalışılmıştır.) ve programların "ikili flat dosyası" denilen ve "bFLT" olarak adlandırılan yapıya dönüştürülmesi ihtiyacı ortaya çıkmıştır.

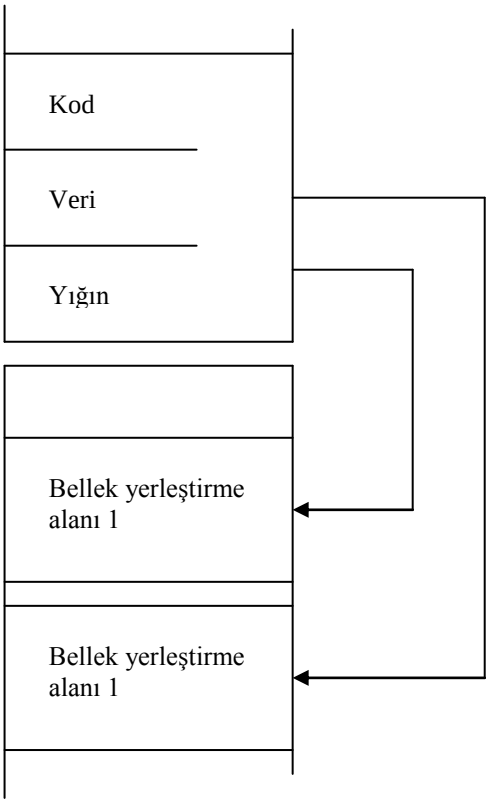
µClinux/Coldfire üzerindeki uygulamalar "flat dosyası" denilen yeni bir yapıda saklanmaktadır. "flat dosyası", çok basit nesne dosyası formatıdır. Kısa bir başlık, kod, veri kısmı ve tekrar yerleşim bilgisinden oluşmaktadır. Tekrar yerleşim bilgisi ise tekrar yerleştirilecek kelimeler ve adresleri içermektedir. Bunlar opcodelar, sabitler ve diğer parçaları içermektedir. Flat dosyasında hata ayıklama bilgisi veya sembolleri bulunmamaktadır.

"flat ikili dosyası", "ELF" yapısındaki nesne dosyasının oluşumundan sonra "elf2flat" aracının kullanılması ile oluşmaktadır. Çeviri süreci çok basittir. Giriş olarak sadece oluşturulmuş "ELF" yapısındaki ikili nesne dosyasını istemektedir. "elf2flat" aracı çeşitli seçeneklere sahiptir. Uygulamanın yığın boyutu tanımlanabilmektedir.

4.2.5 Bellek Dizimi

µClinux'ta uygulamalar 4 farklı bellek bölgesinden oluşmaktadır. Yazı ve kod bölgesi derleyici tarafından üretilen esas komutları içermektedir. Veri bölgesi 2 alt kısımdan oluşmaktadır. Bunlardan birisi, başlangıç değeri verilmiş verilerin

bulunduğu bölge diğeride başlangıç değeri verilmemiş verilerin tutulduğu bölgedir. Bu bölgeye "bss" bölgesi denmektedir. En son bölge ise küme ve program yığınıdır. Şekil 4.2'de bellek bölgeleri görülmektedir.



Şekil 4.2- Bellek bölgeleri

Kod kısmı ve başlangıç değerli veri kısmı, ikili nesne dosyasına yerleştirilmektedir. "bss" kısmı ise boyut bilgisi olarak "flat" ikili dosya başlığına yerleştirilmektedir. Yükleme ve çalışma zamanında çekirdek, "bss" kısmı için yeterli miktarda bellek alanı ayırmaktadır ve bu bölgeyi sıfırla doldurmaktadır.

Küme ise, program yürütümü sırasında dinamik olarak daha fazla bellek kullanımı için oluşturulan bölgedir. µClinux'taki yığın ise program çalışması sırasında oluşturulur ve sabit uzunluktadır. Sanal bellek sistemlerinde, istenildiği zaman küme ve yığın dinamik olarak büyüyebilmektedir. Ancak µClinux'ta yığın istenildiği zaman büyüyememektedir. Bununla beraber küme, eğer çekirdek istenilen uygun bir bellek alanına sahip ise dinamik olarak büyüyebilmektedir.

Bellek alanları yerleştirimi için uygulamalar içinde kullanılan "malloc" isimli standart kütüphane çağrısıdır. Bu, µClinux "mmap" isimli sistem çağrısı kullanılarak gerçekleştirilmektedir.

Çekirdek, “mmap” çağrsı vasıtasıyla yapılan bellek isteklerini tuttuğu boş bellek havuzundan karşılamaktadır. İstenilen bu bellek alanları kullanımı bittikten sonra “munmap” isimli sistem çağrısı ile çekirdeğe geri verilebilmektedir. Çekirdek, her süreç için o sürece ait “mmap” bölgelerinin bir listesini tutmaktadır. Herhangi bir süreç sona erdiği zaman çekirdek ilişkisel sürece ait “mmap” bölgelerini boş bellek havuzuna dahil etmektedir. Bu durumda da “mmap” çağrısının kullanımının avantajı görülmektedir. Çekirdek, program tamamlandığın ya da programdan çıkıldığında o programa ait “mmap” nesnelerini serbest bırakmaktadır.

“malloc” bölgeleri herhangi uygun bellek alanında bulunabilmektedir. Dolayısıyla programın bellek alanı dışında da yer alabilmektedir. Bu işlem uygulamadan bağımsızdır.

4.2.6 Uygulama Derlenmesi ve Yüklenmesi

Uygulama çalışacağı zaman(standart “exec” sistem çağrısı üzerinden) çekirdek, verinin, yığının ve kodun tutulacağı bütün, tek bir bellek alanı ayırmaktadır. Daha sonra çekirdek, program kodunu ve veri kodunu ikili “flat” dosyasından bu bölgeye kopyalamaktadır. “bss” veri kısmını sıfırlarla doldurmakta, gerekli adreslerin tekrar yerleştirimi yapılmaktadır. Tekrar yerleştirim şemasının dezavantajı ise her program yürütümünde uygulama kodunun kopyasının çıkarılmak zorunda olmasıdır. Bu durum sistem RAM’sinin harcanması anlamına gelmektedir.

Bir uygulama programının derlenmesinden yürütülmesine başlanmasına kadar olan işlemler şu basamaklardan geçmektedir [18]:

i- Kodun “elf dosyası” yapısında derlenmesi

ii- “elf2flt” aracı kullanılarak “flat dosyası”oluşturulması

iii- Programın çalıştırılması

iv- μClinux “flat” yükleyicisinin(binfmt_flat) program için bellek alanı ayırması

v- “bss” kısmının sıfırlar ile doldurulması

vi- Yüklenmiş kod üzerinde “tekrar yerleşim” uygulanması

vii- Programın başlangıç koduna atlanılması.

4.2.7 Araçlar

μClinux ile kullanılan araç zinciri, GNU ailesi geliştirme araçları temellidir. Kaynak kod çapraz derleyici kullanılarak derlenmektedir. Derleyici ve diğer ilişkisel araçlar kaynak sistemler üzerinde çalışmaktadır. Bu sistemlerde genellikle Intel tabanlı üzerinde Linux koşan kişisel bilgisayarlardır.

Sistem derlenmeden önce derlemenin hangi sistem için yapılacağı belirlenmektedir. Projede kullanılan Coldfire 5272 işlemcisi olduğundan, bu işlemciye uygun seçenek seçilmektedir. "M68k-tools" çapraz derleyicisi ile "elf" yapısında dosyalar oluşmakta, bunlar "elf2flt" aracı ile "flat" yapısına çevrilmektedir. Sistem geliştirmede kullanılan araçları şu şekilde sıralanır:

i- "M68k-tools" çapraz derleyicisi

ii- "elf2flt" aracı

iii- MetroWorks CodeWarrior Inc. :

Windows 98 üzerinde çalışan, görsel bir platform sunan, kodlar derlenebilmektedir. Windows 98 üzerinden paralel port aracılığı ile geliştirme kiti 5272C3'e BDM arayüzü arasında bağlantı kurulabilmekte ve bu şekilde kodlar yürütme sırasında birebir takip edilebilmekte, hatalar tespit edilebilmektedir.

iv- Motorola kalıcı bellek yükleyicisi- "CFFLASHER":

Bu program Windows 98 üzerinde çalışmaktadır. Programda görüntünün yükleneceği geliştirme kiti modeli ve görüntünün yükleneceği adres seçilebilmektedir. Görüntü, seri port ve BDM arayüzü aracılığı ile sistem kalıcı belleğine yüklenebilmektedir.

4.2.8 Hata Giderme

Hata gidermek için en önemli araçlardan biri ise Coldfire işlemcisinde bulunan geri planda BDM arayüzüdür. BDM, sisteme direkt olarak özel bir donanım girişi sağlamakta ve bu şekilde hata ayıklama yapılabilmektedir. BDM arayüzü birkaç pin ucundan oluşmakta ve CPU'nun BDM modülüne bağlanmaktadır. BDM arayüzü sayesinde kişisel bilgisayara paralel portu aracılığıyla bağlanılabilmektedir. Ayrıca BDM arayüzü sayesinde kalıcı belleğe direkt olarak ikili kod yüklenilebilmektedir.

4.3 μClinux'un Seçilme Nedenleri

Linux, bellek yönetim birimi olmayan ortamlar için çok iyi bir işletim sistemi oluşturmaktadır. μClinux, Linux'tan adapte edildiği için ortak amaçlar ve avantajlar taşımaktadır. Bunlar:

i- Düşük maliyetli, belli bir düzeye gelebilmiş, güvenilir ve kararlı bir işletim sistemidir.

ii- İşletim sistemine yeni özellikler sürekli olarak kazandırılmakta var olan özelliklerin sorunları giderilmekte ve geliştirilmektedir.

iii- μClinux ve Linux'un çok geniş ölçekli geliştirici, programcı ve kullanıcı ağına sahip olmasıdır. Gerek İnternet üzerindeki gruplar gerekse yazılı çıkan dergiler ve üniversitelerdeki öğrenci grupları Linux'un gelişmesine ve Linux'a yeni özellikler kazandırılmasını sağlamakta ve kullanıcılara destek olmaktadır.

iv- Çok geniş kapsamlı cihaz, dosya sistemi ve ağ protokolleri desteği bulunmaktadır.

v- İşletim sisteminin kaynak kodu tamamiyle görülebilmektedir. Bu doğrultuda işletim sistemi istenilen özelliklere ve koşullara göre özelleştirilebilmektedir.

vi- Linux için yazılmış uygulamalar çok fazla platform aktarımı eforu gerekmeden yeni platformlara taşınabilmekte, özellikle μClinux ortamına çok rahatlıkla taşınabilmektedir.

vii- μClinux, çekirdek boyutunun 512 kilosekizliden az ve tüm görüntünün 900 kilosekizliden az olması kaynak sınırlaması olan gömülü sistemler için önemli bir işletim sistemi teşkil etmektedir.

viii- μClinux, sahip olduğu dosya sistem desteği(MSDOS, NFS, doğal EXT2, SMB, ROM ve kalıcı bellek) ve geniş ağ özellikli desteği ile de önemli bir avantaj teşkil etmektedir.

5. PROJENİN UYGULANMASI

5.1 Motorola Coldfire İşlemcisi

Motorola, değişken uzunluklu RISC teknolojisi konseptine dayanarak, Coldfire işlemcilerin, klasik 32-bit sabit uzunluklu RISC mimarisinin basitliği ile bellek kullanımını azaltan değişken uzunluklu komut kümesini birleştirdiğini belirtmiştir.

Coldfire işlemci ailesi, gömülü sistemler pazarı için geliştirilmiştir. Fiyatı 5$ ile 20$ arasında değişmektedir. İşlemci hızı 10 ile 70 MIPS arasında değişmektedir. Coldfire işlemcilerin tam olarak RISC mimarisine sahip oldukları söylenemez. Komut kümesi Motorola 68000 ailesinin bir alt kümesidir. Coldfire mimaride, çok sık kullanılmayan ve karmaşık adresleme metodları çıkarılmıştır. Saklayıcı kümesi ve işlemci durumu, 68000 ailesi işlemcilere benzemektedir. Birçok komut 1 saat çevriminde yürütülebilmektedir. Coldfire işlemcileri, gömülü 68000 ve türevlerinden daha iyi performans sunmaktadır. Bütün Coldfire işlemcileri, dış devre ihtiyacını azaltmak için tümleşik, yonga içinde destek ve arayüz birimleri içermektedir. Bunlardan bazıları seri portlar, DRAM denetleyicileri, zamanlayıcılar, BDM, FEC'dir [7,19,20].

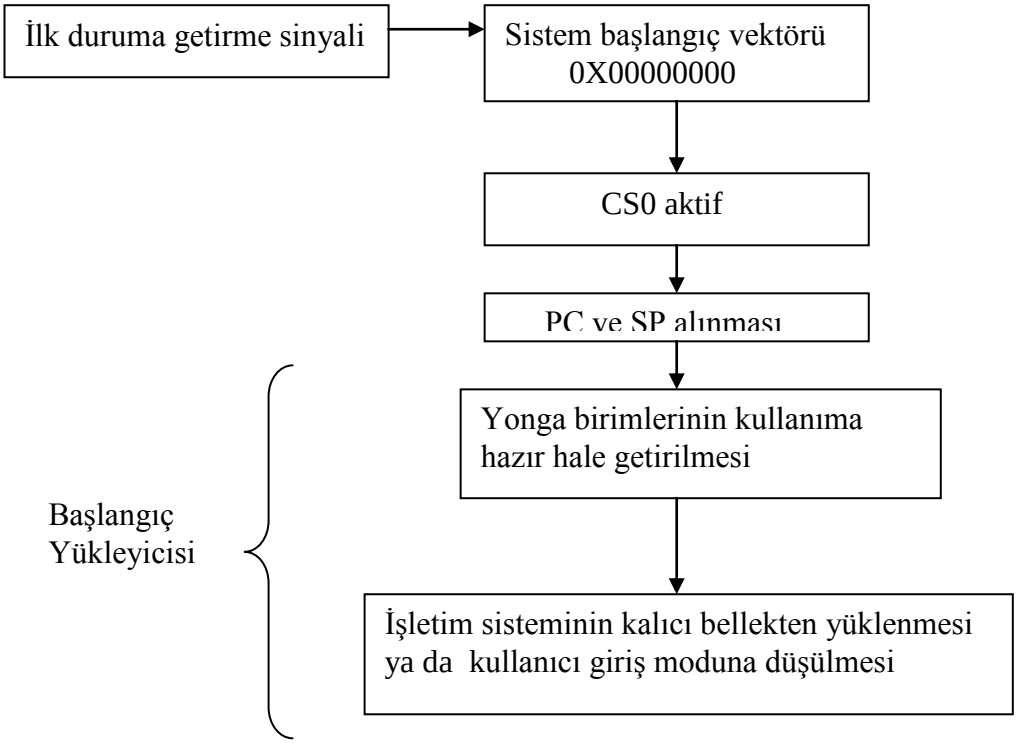
5.2 Geliştirme Kitinin Yapısı

Sistemde temel olarak Motorola 5272C3 geliştirme kiti kullanılmıştır. Uygulamalar bu kit üzerinde geliştirıldikten sonra amaca yönelik bir kart tasarlanmıştır. Bu kart, geliştirme kitinin sadeleşmiş halidir. Şekil 5.1'de Motorola 5272C3 geliştirme kiti görülmektedir [20].

Şekil 5.1– Motorola 5272C3 Geliştirme kiti

Sistemde, başlangıç yükleyicisinin ve sistem görüntüsünün bulunduğu 2 Megasekizli kalıcı bellek ve başlangıç yükleyicisi tarafından bu görüntünün kopyalanıp çalıştırıldığı 4 Megasekizlik bir RAM bulunmaktadır. Sistemde Motorola Coldfire 5272 işlemcisi kullanılmaktadır.

Geliştirme kitinde, ağ bağlantısı için 1 Ethernet portu, 2 seri arayüz vardır. Seri arayüzlerden birisi sisteme direkt erişim için konsol amaçlı kullanılmakta diğeri ise internete erişim için modem bağlantısında kullanılmaktadır. Tasarlanan kartta ise 1 Ethernet portu ve 1 seri arayüz vardır. Şekil 5.2'de, "sistem başlatım süreci" görülmektedir.



Şekil 5.2- Sistemin başlangıç blok diyagramı

Sisteme enerji verildiğinde veya tekrar başlatım düğmesine basıldığında, ilk duruma getirme sinyali aktif olmaktadır. Sistem başlatım vektörü 0x00000000 değerini almaktadır. İşlemci 0x00000000 adresinden PC ve SP'yi okumaktadır. Başlangıçta CS0 aktif olduğundan ve CS0 kalıcı belleğe bağlı olduğundan okuma işlemi kalıcı bellekten yapılmaktadır. Daha sonra sistem birimleri kullanıma hazır hale getirilmektedir. Örnek olarak UART modüllerinin, yonga içindeki belleğin kullanıma hazır hale getirilmesi verilebilmektedir. Daha sonra başlangıç yükleyicisi kalıcı bellekteki işletim sistemi görüntüsünü RAM'ye yüklemektedir. Eğer kullanıcı başlangıç yükleyicisi işletim sistemini yüklemeden "Esc" tuşuna basarsa sistem "kullanıcı giriş" moduna düşmektedir. Bu modda kullanıcı "tftp hizmetlisi"ni

kullanarak istediği makinadan yeni bir işletim sistemi görüntüsünü belleğe yükleyebilmektedir. Bu görüntü RAM'nin 0x20000 adresine yazılmaktadır. Belleğe yükleme yapıldıktan sonra "go 20000" komutu ile işletim sistemi çalıştırılabilmektedir.

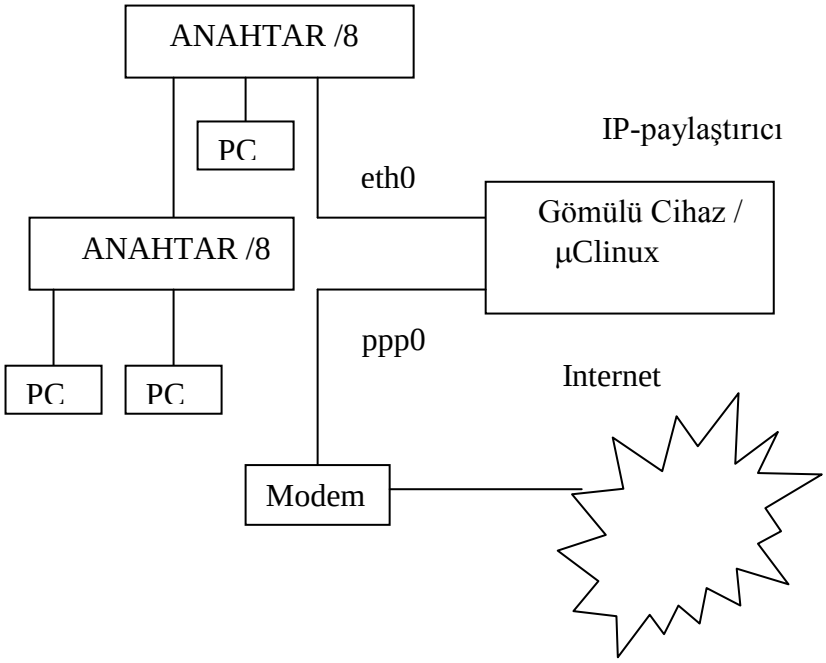
5.3 Sistem Fonksiyonları

Sistem, IP-paylaştırıcı, IP-güvenlik duvarı ve IP-yönlendirici işlevlerini yerine getirmek üzere kurulmuştur. Bununla ilgili μClinux çekirdek ayarları yapılmış ve gerekli yazılımlar sisteme yüklenmiştir. Sistemde Linux 2.0.38 çekirdek uyarlaması olan μClinux koşmaktadır.

5.3.1 IP-Paylaştırıcı

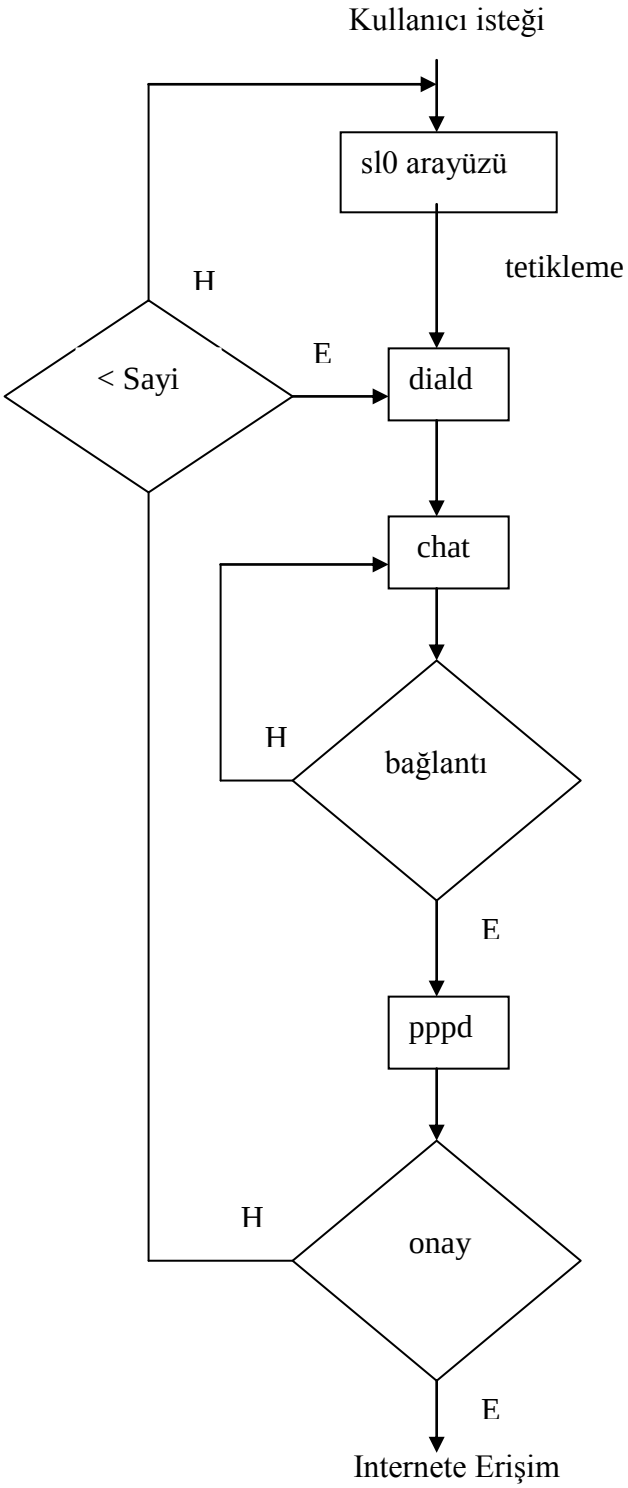
IP paylaştırma, tek bir geçerli IP üzerinden birçok makinanın internete bağlanabilmesini sağlamaktır.

Sistemin bu işlevi yapabilmesi için öncelikle çekirdeğin TCP/IP desteği açılmalıdır. Daha sonra "IP Güvenlik Duvarı" özelliği ve onun altında bulunan "IP Maskeleme" özelliği açılmalıdır [21]. Bu özellik ile sisteme eklenecek güvenlik duvarı kuralları ile IP maskeleme yapılabilmektedir. İnternet bağlantısı seri arayüz tarafından sağlanmaktadır. Seri arayüz aracılığı ile modem bağlantısı yapılmakta ve servis sağlayıcıya ulaşılmaktır. Seri arayüz üzerinden "PPP protokolü" konuşulmaktadır. Sistemin içinde bulunduğu çalışma ortamı Şekil 5.3'te görülmektedir.



Şekil 5.3- Sistemin çalışma ortamı

Şekil 5.3'te, anahtarlara bağlı bilgisayarlar Internete, IP-paylaştırıcı üzerinden ulaşmaktadırlar. Şekil 5.4'de IP-paylaştırıcı üzerinde, kullanıcının istek yapmasından servis sağlayıcıya bağlanılması sürecine kadar olan işlemler blok diyagram olarak gösterilmiştir.



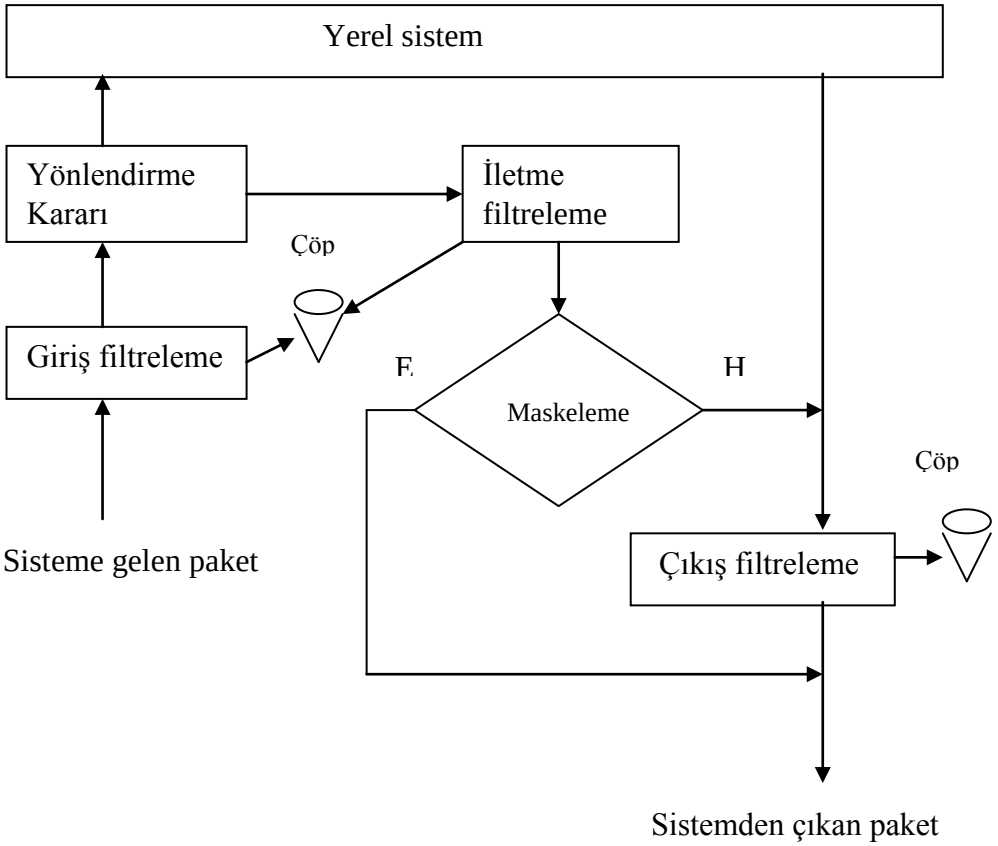
Şekil 5.4- Kullanıcı istek-internete bağlantı süreci blok diyagramı

Kullanıcı internete erişim isteğinde bulunduğu zaman, bu istek “diald” isimli programın açtığı “SLIP” arayüzüne yönlendirilmektedir. “diald”, bu arayüze gelen isteği tespit ettiği anda “chat” isimli programı çalıştırmaktadır. “chat” programı modemle iletişim için kullanılmaktadır. Servis sağlayıcı ile fiziksel bağlantı kurulduktan sonra “diald” tarafından “pppd” isimli program çalıştırılmaktadır. “pppd” programı servis sağlayıcı ile PPP protokolü haberleşmesini sağlamaktadır. Servis sağlayıcıyla internet bağlantısı yapılamadığı zaman ya da servis sağlayıcı internet erişimi için onay vermediği zaman önceden belirlenen erişim deneme sayisina(<Sayi) göre servis sağlayıcı tekrar tekrar aranmaktadır. Sayı aşıldığı zaman kullanıcıdan tekrar internete erişim isteği beklenmektedir.

5.3.2 IP-Güvenlik Duvarı

IP-güvenlik duvarı, TCP/IP katmanına göre gelen, giden ve iletilen paketlerin filtrelenme işlemini yapmaktadır.

Sistemin bu işlevi yapabilmesi için TCP/IP ağ desteğine ek olarak “IP Güvenlik Duvarı” özelliğininde çekirdeğin derlenmesi aşamasında açılması gereklidir. Sistemin “/proc/net” dizini altında çekirdeğin yönlendirme, güvenlik duvarı dosyaları vardır. Çekirdek bu dosyalara göre IP paketlerini yönlendirmekte ve filtrelemektedir. Çekirdek, yönlendirme için “route”, gelen paketlerin filtrelenmesi için “ip_input”, giden paketlerin filtrelenmesi için “ip_output”, yönlendirilen paketler için “ip_forwarding” isimli dosyaları kullanmaktadır. Yine “Route” isimli program ile çekirdeğin yönlendirme tablosu, “ipfwadm” isimli program ile çekirdeğin paket filtrelenmesinde kullanılan “ip_input”, “ip_output”, “ip_forwarding” isimli dosyaları içinde bulunan tablolar değiştirilmektedir. Şekil 5.5’de sisteme gelen bir paketin sistemde izlediği yol görülmektedir.



Şekil 5.5- Gelen paketin sistemde izlediği yol

Sisteme herhangi bir arayüzünden(eth0 veya ppp0) bir paket geldiği zaman öncelikle giriş filtrelemesinden geçmektedir. Gelen paket, "ip_input" dosyası içindeki filtreleme tablosu ile karşılaştırılmakta ve pakete ne yapılacağı karar verilmektedir. Eğer paketin geçişine izin verilirse, paket için yönlendirme kararı verilmektedir. Eğer paket yerel sisteme gelmişse(hedef internet adresi sistemin internet adresi ise) sistemdeki ilgili uygulamaya yollanmaktadır. Paket sistem içindeki bir uygulama için değilse "route" dosyası içindeki yönlendirme tablosuna bakılmakta ve yönlendirme kararı verilmektedir. Yönlendirme kararı verildikten sonra, paket yönlendirme filtrelenmesinden geçmektedir. Bunun için "ip_forwarding" dosyası içindeki iletme tablosundaki kurallar ile karşılaştırılmaktadır. Eğer paket iletme tablosundan da geçerse ve paketin maskelenmesine karar verilmişse, direkt hedef arayüz üzerinden ağ üzerine çıkarılmaktadır. Eğer paket maskelenmeyecekse çıkış filtresinden geçmektedir. Bunun için "ip_output" dosyası içindeki çıkış tablosundaki kurallar ile karşılaştırılmaktadır. Bu kurallara göre paketin ilgili arayüzden çıkıp çıkmayacağına karar verilmektedir.

5.3.3 IP-Yönlendirici

IP-yönlendirici, sisteme gelen IP paketleri için çekirdek yönlendirme tablosuna göre yönlendirme işlemi yapmaktadır. "Route" isimli program ile çekirdek yönlendirme tablosu değiştirilebilmektedir [21].

Sistemin bu işlevi yapabilmesi için TCP/IP ağ desteğine ek olarak "IP İletme" özelliğininde çekirdeğin derlenmesi aşamasında açılması gereklidir. Yönlendirme işlemi, sistemde yönlendirme protokolleri çalışmadığından dinamik olarak yapılamamaktadır. Şekil 5.6'da yönlendirmede kullanılan temel yapı gösterilmektedir.

Hedef IP	Alt ağ maskesi	Geçityolu	Arayüz

Şekil 5.6- Yönlendirmede kullanılan temel yapı

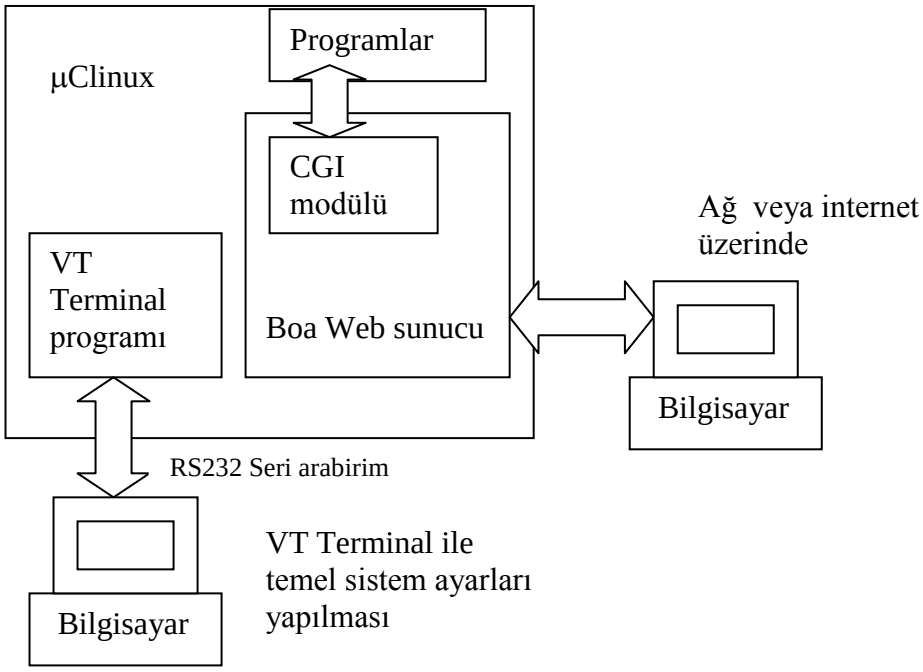
Sistemde bir paketin yönlendirilirken "/proc/net/" dizini altındaki "route" isimli dosya içindeki yönlendirme tablosuna bakılmaktadır. Bu tabloya yine "Route" isimli program kullanarak eklemeler veya çıkartmalar yapılabilmektedir. Bir paketin yönlendirilebilmesi için temel olarak bir yapıya ihtiyaç vardır(Şekil 5.6). Gelen paket, bu temel yapıdaki alt ağ maskesiyle "ve" işleminden geçmektedir. Çıkan sonuç "hedef IP" ile karşılaştırılmaktadır. Eğer sonuç aynı çıkarsa, paket belirtilen geçityoluna yine belirtilen arayüz üzerinden yollanmaktadır.

5.4 Sistemin Web Üzerinden Yönetilmesi

Sistemin temel ayarları olan IP adresi, alt ağ maskesi, geçityolu adresi verme işlemleri sistemin RS232 seri arabirimi üzerinden yapılabilmektedir. Ancak IP paylaştırma, IP yönlendirme tablosu, IP güvenlik duvarı ile ilgili ayarlar web üzerinden yapılabilmektedir.

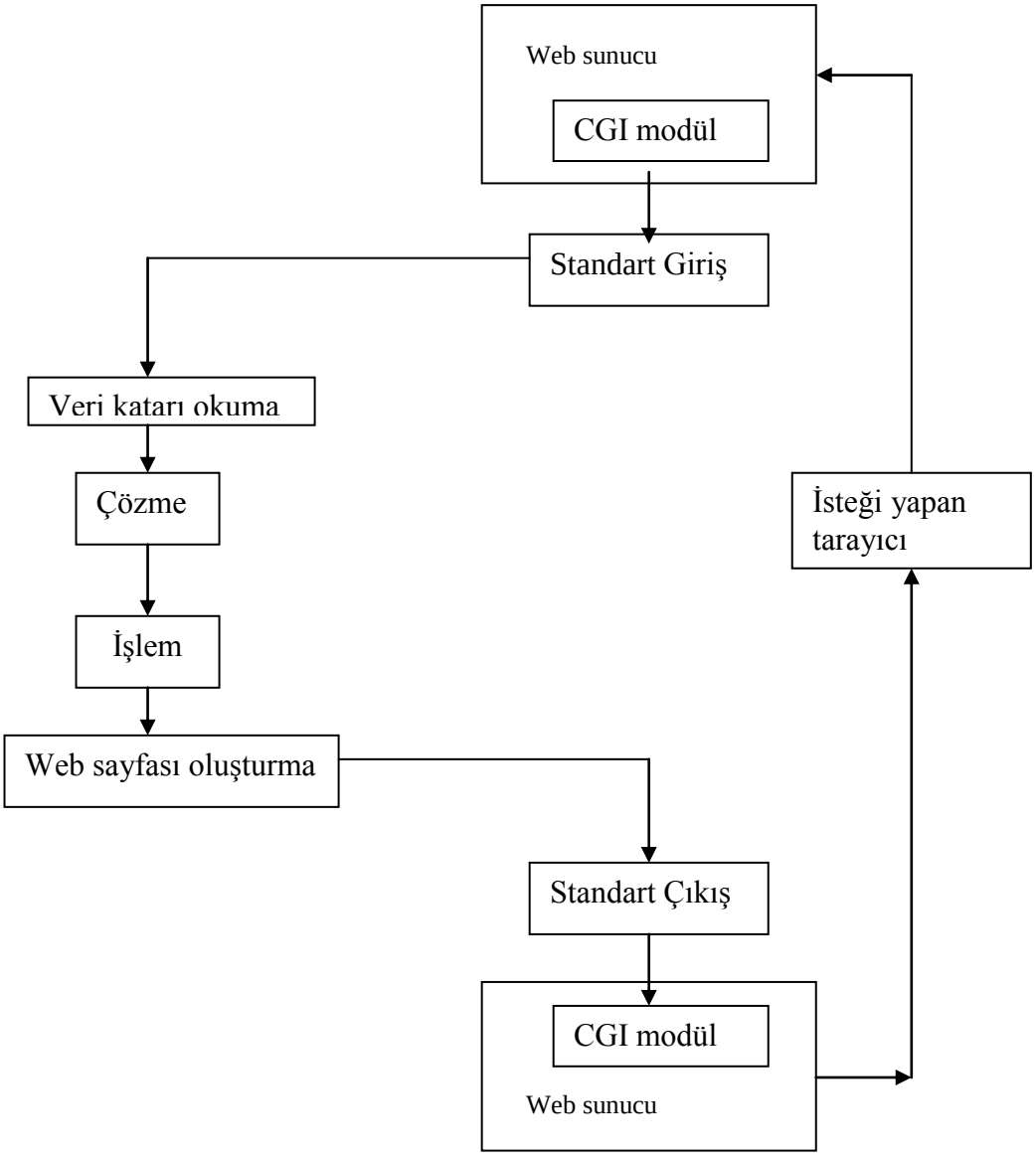
Sistemin web üzerinden ayarlarının yapılabilmesi için öncelikle sistem üzerinde çalışabilen web sunucu seçilmesi gerekmektedir. Kapladığı bellek alanı ve CGI desteğinden dolayı sunulan alternatifler içinde en uygun olarak "BOA" isimli web sunucusu seçilmiştir.

BOA web sunucu içindeki CGI modülü sayesinde sistemde bulunan kodlar, web aracılığı ile aktarılan parametrelere göre çalışmaktadırlar. Sistemin temel ayarlarının yapılması ve web üzerinden çalışması Şekil 5.7'de gösterilmektedir.



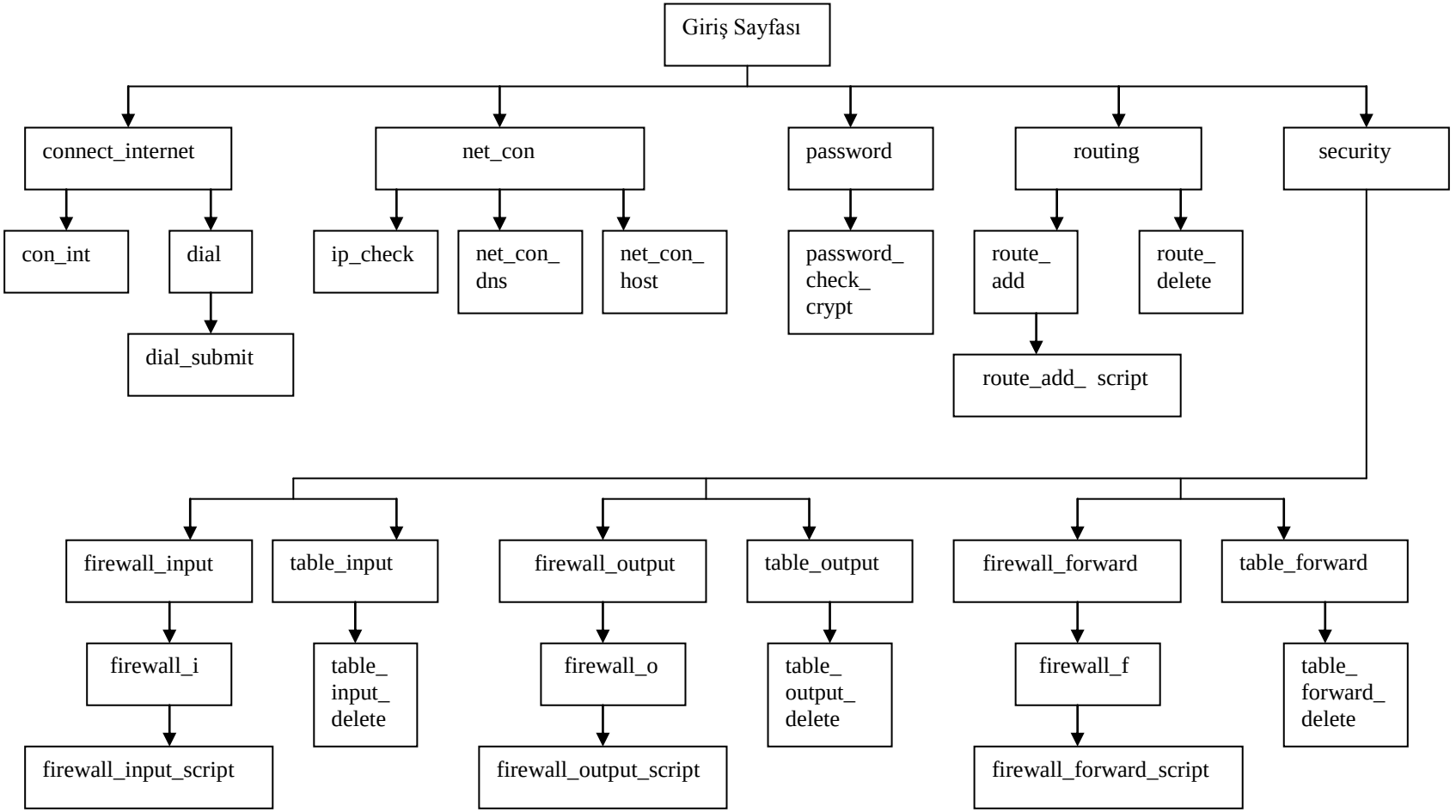
Şekil 5.7- Sistemin temel ayarlarının yapılması ve web üzerinden çalışması

Sistemin web üzerinden yönetimi için yapılan web sayfaları giriş sayfası dışında "C" programlama dili ile yazılmış CGI programlarıdır. Programların genel çalışma yapısı Şekil 5.8'de gösterilmiştir. CGI çalışma yapısı konusunda örneklerden faydalanılmıştır[22]. Programların yazımında kullanılabilecek kütüphane fonksiyonlarından da öğrenme amaçlı faydalanılmıştır [23].



Şekil 5.8- Programların genel çalışma yapısı

Ağ veya internet üzerindeki bir kullanıcı CGI porgramları aracılığıyla sistemde değişiklik yapmak istediğinde sisteme bu değişiklikle ilgili bilgi ya da parametreler yollamaktadır. Boa web sunucusunun CGI modülü bu bilgi ve parametreleri alarak sistemin standart girişine veri katarı olarak yollamaktadır. CGI programları standart girişi okuyarak bu veri katarlarını almaktadırlar. Veri katarları, içindeki faydalı bilginin elde edilmesi için bir çözme sürecinden geçmektedir. Elde edilen faydalı bilgi ve parametreler ile yapılması gereken işlemler yapılmaktadır. Elde edilen sonuçlar ile ilgili web sayfası oluşturularak standart çıkışa yollanmaktadır. CGI modülü, web sayfasını standart çıkıştan okumakta ve web sunucusuda web sayfasını isteği yapan web tarayıcısına yollamaktadır. Giriş sayfası dışında "C" programlama diliyle yazılmış CGI kodları ağaç yapısı olarak Şekil 5.9'da verilmiştir.



Şekil 5.9- CGI programlarının ağaç yapısı olarak gösterilimi

Ağaç yapısı içindeki programların fonksiyonlarını açıklamak gerekirse;

connect_internet: İnternete bağlanmak için gerekli ayarların girileceği web sayfasını getirmektedir.

con_int: "connect_internet" sayfasında girilen parametreleri alarak sistemde gerekli değişiklikleri yapmaktadır.

dial: İnternet bağlantısının sürekliliği ve telefon çevirme özellikleri ile ilgili ayarların yapılacağı web sayfasını getirmektedir.

dial_submit: "dial" sayfasında girilen parametreleri alarak sistemde gerekli değişiklikleri yapmaktadır.

net_con: Sistemin internet adresinin, DNS sunucularının ve sistemin isminin ayarlanabildiği web sayfasını getirmektedir.

ip_check: Sistemin girilen yeni internet adresini kalıcı bellekteki yerine yazmaktadır.

net_con_dns: Sistemin DNS sunucularını kalıcı bellekteki yerine yazmaktadır. Aynı zamanda "/etc/config/resolv.conf" dosyası içinde de gerekli değişiklikleri yapmaktadır.

net_con_host: Sistemin yeni ismini kalıcı bellekteki yerine yazmaktadır.

password: Sistemin web üzerinden erişim şifresini değiştirmek için gerekli sayfayı getirmektedir.

password_check_crypt: Sistemin web üzerinden giriş şifresini değiştirmekte ve şifreyi kalıcı belleğe yazmaktadır.

routing: Sistemin yönlendirme tablosunu ve sistem arayüzlerine ilişkin bilgileri göstermektedir. Ayrıca sisteme yönlendirme kuralları eklemek veya çıkarmak için bir arayüz sunmaktadır.

route_add: Yönlendirme tablosuna yeni kural eklenebilmesi için "/var/config/" dizini altındaki "route_add" isimli dosyaya gerekli komut satırını yazmaktadır. Aynı zamanda kalıcı belleğe, hedef adres, alt ağ maskesi, geçityolu ve arayüz bilgileri yazılmaktadır.

route_add_script: "route_add" isimli dosyayı çalıştırarak yeni kuralın yönlendirme tablosuna eklenmesini sağlamaktadır.

route_delete: Yönlendirme tablosundan kural silmektedir. Aynı zamanda kalıcı bellekten silinen kuralla ilgili bilgileri kaldırmaktadır.

security: Güvenlik duvarı ayarlarına geçiş yapılmasını sağlayan sayfayı getirmektedir.

firewall_input: Sistemin giriş filtrelemesinde kullanılan kurallara ek yapılabilmesi için gerekli web sayfasını getirmektedir.

firewall_i: Sistemin giriş filtresine eklenecek kuralı "/var/config" dizini altındaki "firewall" isimli dosyaya yazmaktadır.

firewall_input_script: "firewall" dosyasındaki kuralı çalıştırmaktadır. Aynı zamanda kalıcı belleğe bu kuralı uygun yapıda yazmaktadır.

table_input: Giriş filtrelemesinde kullanılan kuralları göstermektedir. Aynı zamanda kural veya kuralların silinmesi için gerekli arayüzü oluşturmaktadır.

table_input_delete: Giriş filtrelenmesinde kullanılan kuralların bir veya birkaçının silinmesini sağlamaktadır. Ayrıca silinen kural veya kuralların kalıcı bellekten kaldırılmasını da yapmaktadır.

firewall_output: Sistemin çıkış filtrelemesinde kullanılan kurallara ek yapılabilmesi için gerekli web sayfasını getirmektedir.

firewall_o: Sistemin çıkış filtresine eklenecek kuralı "/var/config" dizini altındaki "firewall" dosyasına yazmaktadır.

firewall_output_script: "firewall" dosyasındaki kuralı çalıştırmaktadır. Aynı zamanda kalıcı belleğe bu kuralı uygun yapıda yazmaktadır.

table_output: Çıkış filtrelemesinde kullanılan kuralları göstermektedir. Aynı zamanda kural veya kuralların silinmesi için gerekli arayüzü oluşturmaktadır.

table_output_delete: Çıkış filtrelenmesinde kullanılan kuralların bir veya birkaçının silinmesini sağlamaktadır. Ayrıca silinen kural veya kuralların kalıcı bellekten kaldırılmasını da yapmaktadır.

firewall_forward: Sistemin iletme filtrelemesinde kullanılan kurallara ek yapılabilmesi için gerekli web sayfasını getirmektedir.

firewall_f: Sistemin iletme filtresine eklenecek kuralı "/var/config" dizini altındaki "firewall" dosyasına yazmaktadır.

firewall_forward_script: "firewall" dosyasındaki kuralı çalıştırmaktadır. Aynı zamanda kalıcı belleğe bu kuralı uygun yapıda yazmaktadır.

table_forward: İletme filtrelemesinde kullanılan kuralları göstermektedir. Aynı zamanda kural veya kuralların silinmesi için gerekli arayüzü oluşturmaktadır.

table_forward_delete: İletme filtrelenmesinde kullanılan kuralların bir veya birkaçının silinmesini sağlamaktadır. Ayrıca silinen kural veya kuralların kalıcı bellekten kaldırılmasını da yapmaktadır.

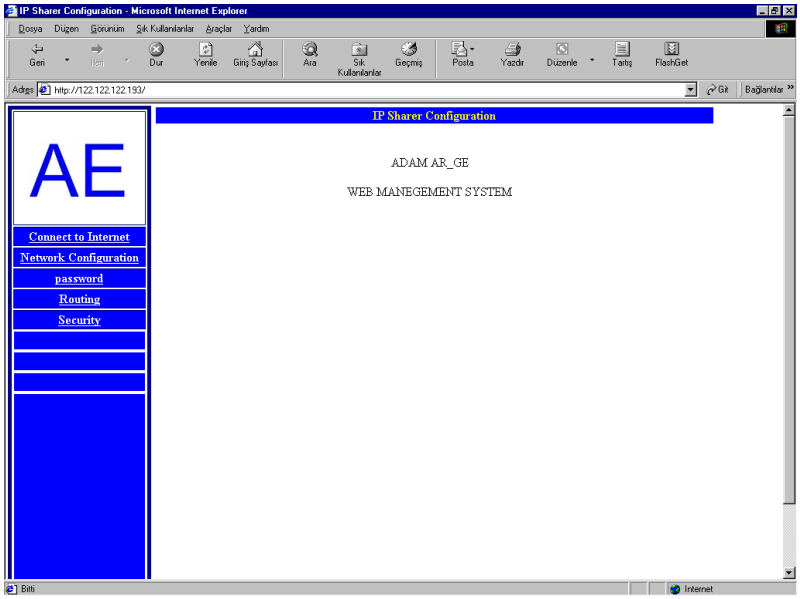
Ağaç yapısı dışındaki programlara ek olarak 3 program daha yazılmıştır. Bunlar:

packet_check: Belirtilen paketlerin geçityolunda nasıl bir davranış biçimi ile karşılaşacağını belirlemektedir. Bunun için kontrol edilecek paketle ilgili kural "/var/config/" dizini altındaki "firewall" isimli dosyaya yazılmaktadır.

packet_check_script: "firewall" dosyası içine yazılmış kuralı çalıştırmaktadır. Sonuç "firewall_check" dosyasından okunmakta ve web sayfası olarak görüntülenmektedir.

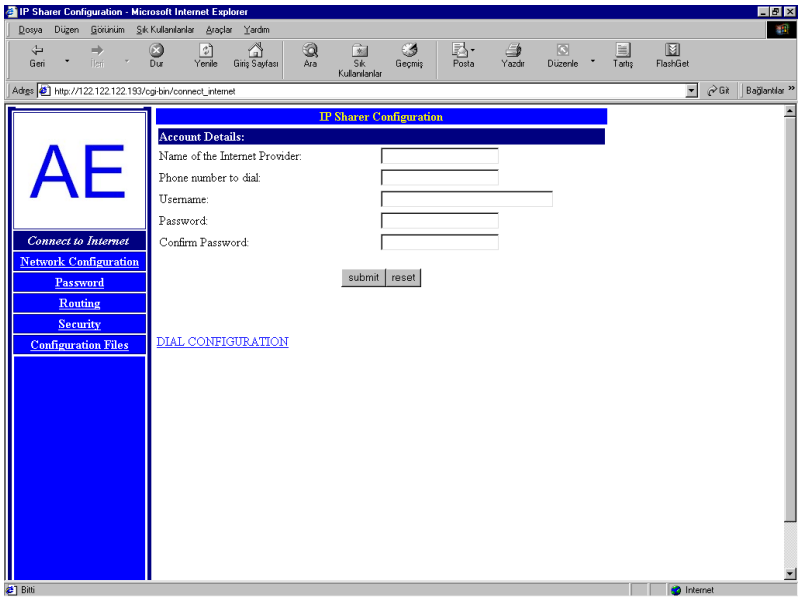
reboot: Sistemin yeniden başlamasını sağlamaktadır. Sistemde bulunan Wacthdog zamanlayıcısını kurmakta ve bu zamanlayıcı tazelenmediği için sistem tekrar başlamaktadır.

Sisteme web üzerinden bağlanıldığı zaman Şekil 5.10'da gösterilen giriş sayfası gelmektedir.



Şekil 5.10- Giriş web sayfası

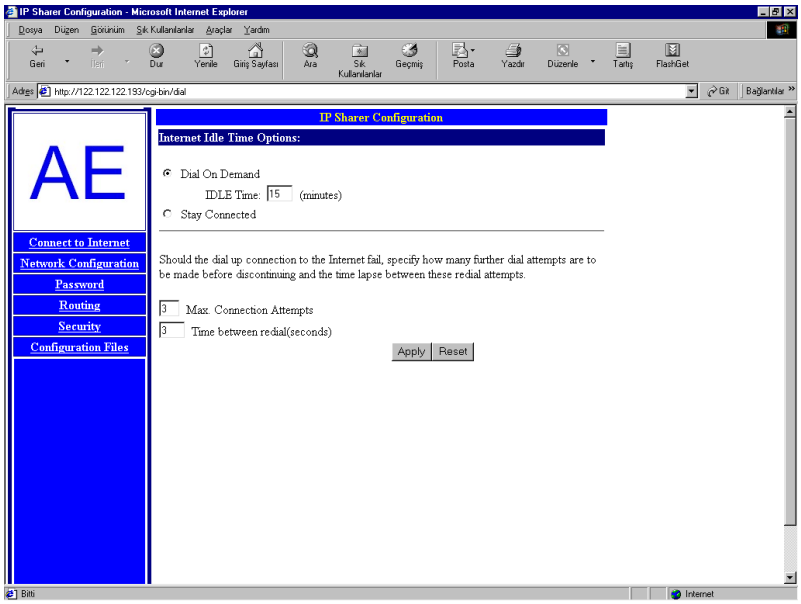
Giriş sayfası üzerinden diğer sayfalara ulaşılabilmektedir. Sistem giriş sayfası HTML koduyla yazılmıştır. Sistemin internete bağlanmak için gerekli ayarların yapıldığı "connect_internet" isimli web sayfası Şekil 5.11'de görülmektedir.



Şekil 5.11- "connect internet" isimli web sayfası

"connect internet" web sayfası üzerinden internet servis sağlayıcının adı, aranacak telefon numarası, kullanıcı adı ve parolası girilmektedir. Bu sayfa üzerinden telefon çevirme ayarlarının yapıldığı sayfaya ulaşılmaktadır.

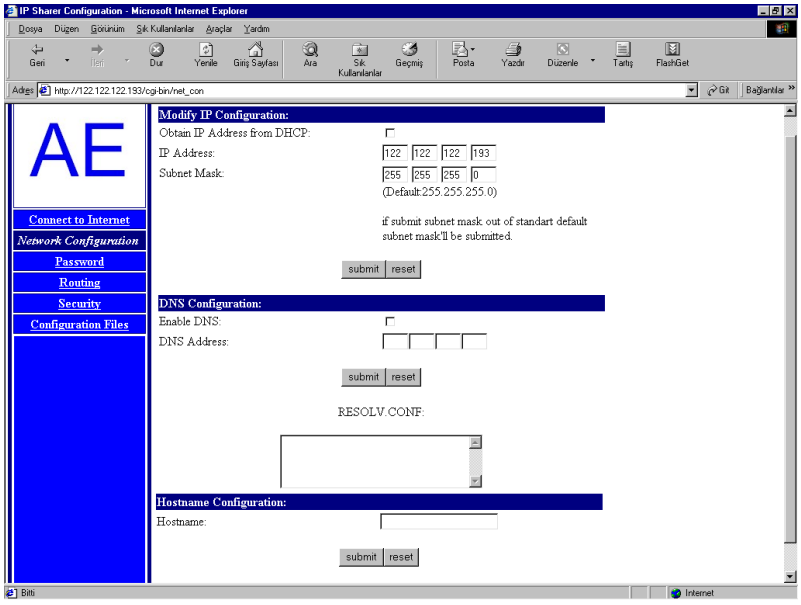
Şekil 5.12'de telefon çevirme ve internet bağlantısı sürekliliği ile ilgili ayarların yapıldığı "dial" isimli web sayfası görülmektedir.



Şekil 5.12- "dial" isimli web sayfası

"dial" web sayfası üzerinden internet bağlıtısının sürekliliği ayarlanabilmektedir. Bu süre en son internete yönlendirilen paketten sonra işlemektedir. Eğer istenirse internet bağlantısı kesilmeden, sürekli olarak kalabilmektedir. Ayrıca en fazla kaç kere telefon numarasının çevirileceği ve bu çevirmeler arasındaki sürede

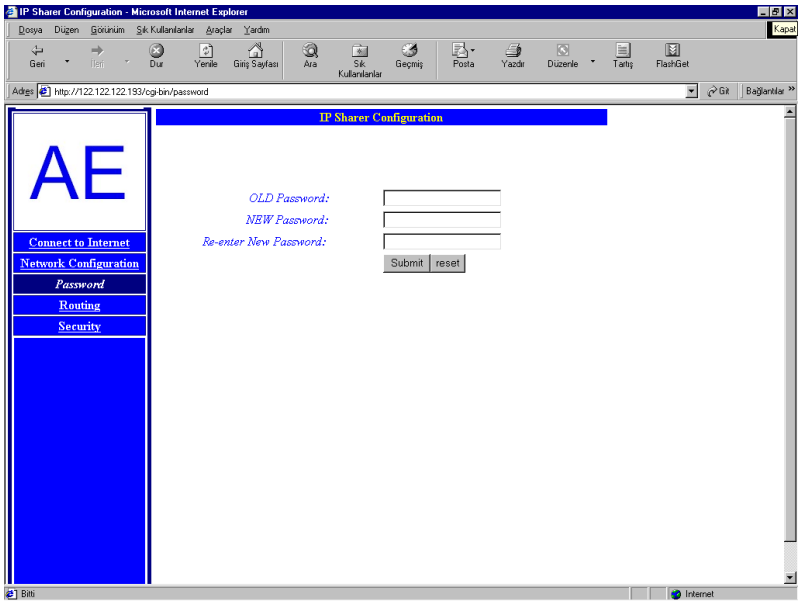
ayarlanabilmektedir. Şekil 5.13'de ağ ayarlarına ilişkin "net_con" isimli web sayfası görülmektedir.



Şekil 5.13- "net_con" isimli web sayfası

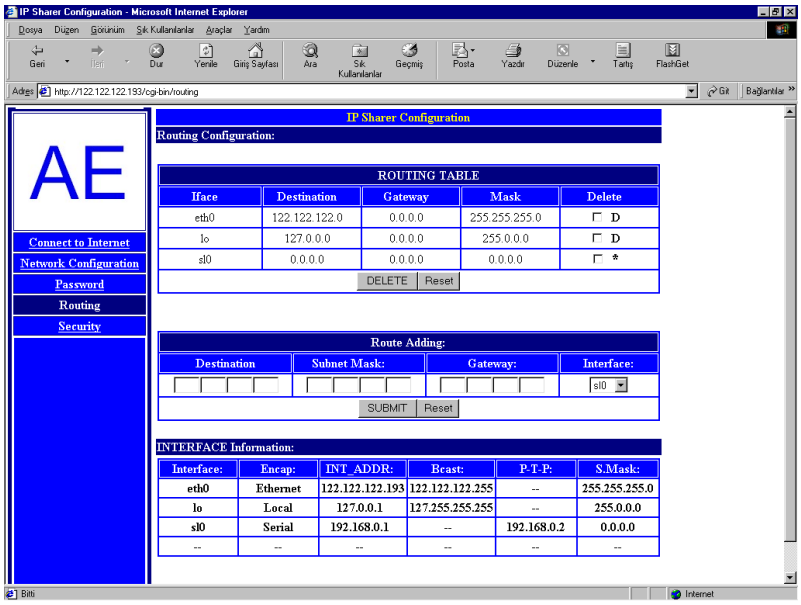
"net_con" web sayfası üzerinden sistemin internet adresi ve alt ağ adresi girilebilmektedir. Sistemin DNS sunucuları belirlenebilmektedir. Aynı zamanda sistemin adı değiştirilebilmektedir.

Şekil 5.14'de web üzerinden erişimde kullanılan parolanın değiştirilebildiği "password" isimli web sayfası görülmektedir.



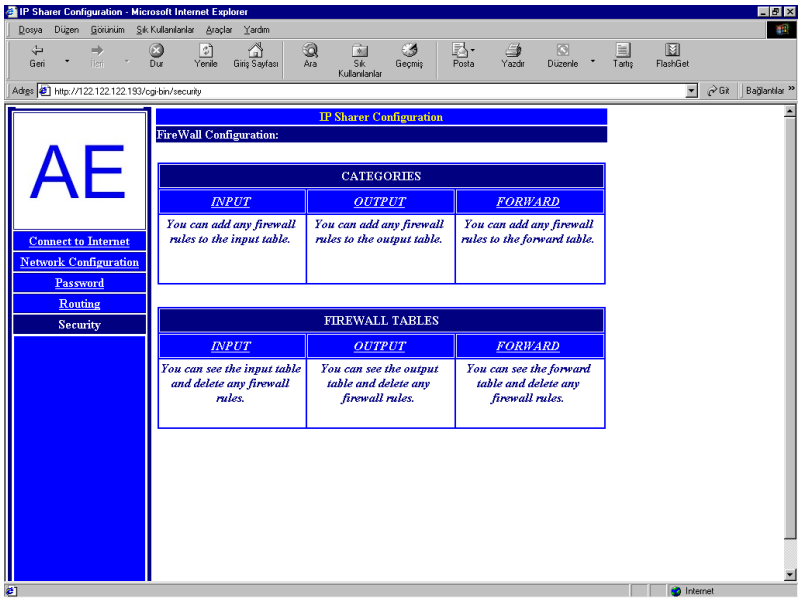
Şekil 5.14- "password" isimli web sayfası

“password” web sayfası ile eski şifre girilmektedir. Daha sonra yeni şifre ve yeni şifrenin tekrarı girilmektir. Şekil 5.15’te sistemin yönlendirme ayarlarının yapıldığı “routing” isimli web sayfası görülmektedir.



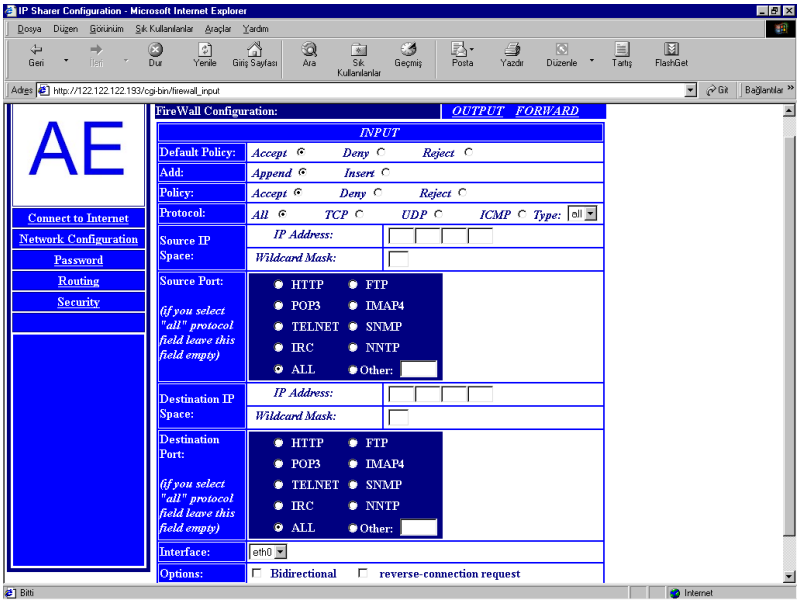
Şekil 5.15- “routing” isimli web sayfası

“routing” web sayfasında sistem yönlendirme tablosu görülmektedir. Yönlendirme tablosuna yeni bir kural ekleme ve çıkarma imkanı sunmaktadır. Kural eklerken hedef internet adres, alt ağ maskesi, geçityolu ve arayüz bilgileri girilmektedir. Bu değerlerden herhangi biri veya birkaçı girilmezse değer olarak sıfır(0) algılanmaktadır. Sayfanın alt kısmında ise sistemdeki arayüzlere ilişkin bilgiler verilmektedir. Şekil 5.16’da sistem güvenliği ile ilgili, “security” isimli ana web sayfası görülmektedir.



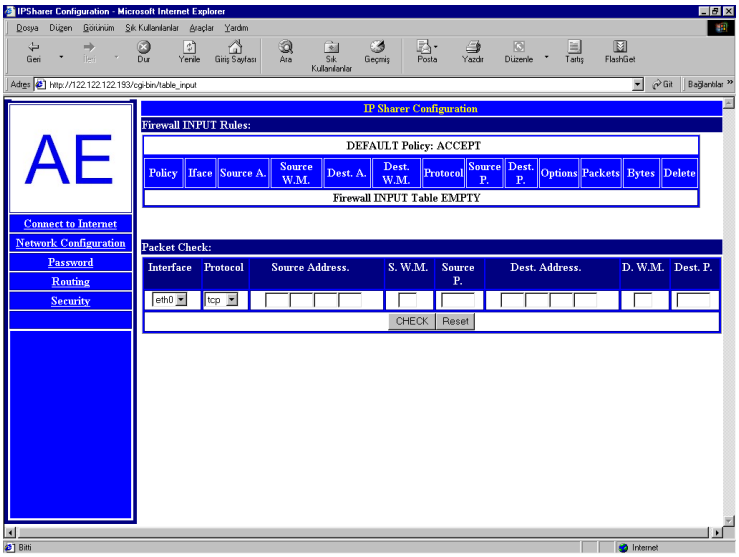
Şekil 5.16- “security” isimli web sayfası

"security" web sayfasında açıklamaları ile beraber güvenlik duvarı ile ilgili kategoriler bulunmaktadır. Bu kategoriler, giriş, çıkış ve iletme olarak belirtilmiştir. Ayrıca güvenlik duvarı tablolarını bu sayfa aracılığıyla görmek mümkündür. Şekil 5.17'de giriş geçityolu tablosuna yeni bir kural eklenmesini sağlayan "firewall_input" isimli web sayfası görülmektedir.



Şekil 5.17- "firewall_input" isimli web sayfası

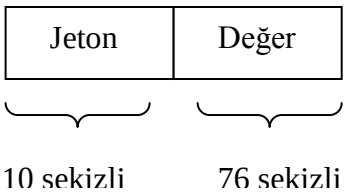
"firewall_input" sayfasında geçityolu tablosuna kural ekleyebilmek için gerekli bilgiler görülmektedir. Bu bilgiler ise gelen pakete nasıl davranılacağı, kuralın tablonun hangi kısmına ekleneceğini, hangi protokol için uygulanacağını, kaynak ve hedef internet adresi uzayını, kaynak ve hedef port numaralarını ve ek seçenekleri içermektedir. Şekil 5.18'de güvenlik duvarı giriş tablosunun görülebildiği "table_input" isimli web sayfası görülmektedir.



Şekil 5.18- "table_input" web sayfası

“table_input” sayfası ile güvenlik duvarı giriş tablosu görülmektedir. Aynı zamanda bu tablodan kural veya kurallar silinmesine imkan vermektedir. Bu sayfa üzerinden bir paketin özelliklerini vererek o paketin giriş filtrelemesinden geçip geçmeyeceği anlaşılabilmektedir. Paket kontrolü için arayüz, protokol, kaynak internet adresi uzayı, hedef internet adresi uzayı girilmektedir.

Sistem ayarlarının kalıcı olabilmesi için gerekli bilgiler kalıcı bellekte saklanmaktadır. Sisteme tekrar başladığında bu ayarlar kalıcı bellekten “startup” isimli program tarafından okunarak sistem üzerinde gerekli değişiklikler yapılmaktadır. Kalıcı belleğe erişim için gerekli “am29xxxx.c” isimli sürücü programı ve kalıcı belleğe yazma, kalıcı bellekten okuma işlemleri için “flashfile.c” isimli program kullanılmaktadır. Sistem ayarları ile ilgili parametreler kalıcı bellekte Şekil 5.19’da ki yapıda saklanmaktadır.



Şekil 5.19- Sistem parametrelerinin kalıcı bellekte saklandığı yapı

Jeton için 10 sekizli, sistem ayarlarının gereklerinden dolayı jetonun değeri için 76 sekizli yer ayrılmıştır. Sistem ayarları için kalıcı bellekte 8 kilosekizli yer ayrılmıştır.

Sistemde bulunan, güvenlik duvarı jetonları dışındaki jeton-değer çiftleri, Tablo 5.1’de örnek değerleriyle gösterilmiştir.

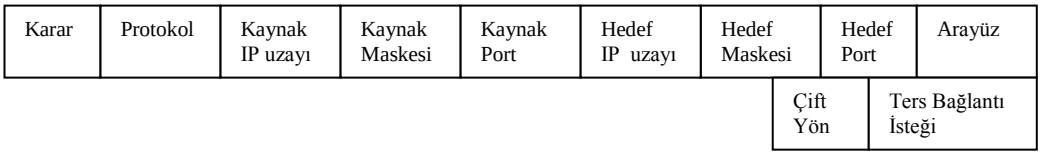
Tablo 5.1’de, “pw_boa” jetonu sistemin “crypt()” algoritması ile şifrelenmiş web erişim parolasını, “isp” jetonu Internet servis sağlayıcısının ismini, “phone” jetonu aranılacak telefon numarasını, “user” jetonu kullanıcı adını, “pw” jetonu kullanıcı şifresini, “idle” sistemin istek olmadan Internet erişimini kesme süresini, “retry-c” jetonu telefon çevirme deneme miktarını, “red-timeo” jetonu deneme girişimlerinin aralık zamanını, “names1” ve “names2” jetonu isim sunucularını, “hostname” jetonu sistemin ismini, “net?”, “mask?”, “gw?”, “int?” jetonları sıra ile bir yönlendirme kuralı için ağ adresini, ağ maskesini, geçityolunu, arayüz bilgilerini(“?” işareti değeri doğal sayılar olan bir değişkeni temsil etmektedir. Bu değişken sistemde bulunan değerlere göre değişmektedir. Örneğin sistemde net0, mask0, gw0, int0’dan net3, mask3, gw3, int3’e kadar jetonlar var ve bir de net5, mask5, gw5, int5 jetonu var. Yeni bir yönlendirme kuralı için jetonlar ekleneceği zaman sistemde varolan jeton değerleri taranmaktadır. Boş bulunan değer için jetonlar yazılır. Bu durumda yeni kural için jetonlar net4, mask4, gw4, int4 olmaktadır. Bu tarama işlemi 0’dan 88’e kadar yapılmaktadır. Çünkü sisteme en fazla 89 jeton-değer çifti

yazılabilmektedir. Bir yönlendirme kuralı silineceği zaman sistemdeki bütün jetonlar taranmakta ve silinecek kural parametreleri ile karşılaştırılmaktadır.) tutmaktadır.

Tablo 5.1 – Sistemdeki jeton-örnek değer çiftleri

JETON	DEĞER
pw_boa	abBqwxv.nBIS5
isp	Superonline
phone	08222112500
user	Onurekinci
pw	123eE56
idle	20
retry_c	5
red_timeo	5
names1	122.122.122.254
names2	122.122.122.253
hostname	Adamarge
net?	160.75.2.20
mask?	255.255.255.255
gw?	122.122.122.254
int?	eth0

Sisteme eklenecek güvenlik duvarı kuralları için bir "jeton değeri" yapısı geliştirilmiştir. Bu yapı Şekil 5.20'de gösterilmektedir.



Şekil 5.20– Güvenlik duvarı kuralları için "Jeton değeri" yapısı.

Şekil 5.20'de, "karar" kısmı kuralın kabul edileceğini(accept), reddedileceğini(deny) veya kabul edilmeyip, kabul edilmedğine dair mesaj yollanacağını(reject), "protokol" kısmı kuralın hangi protokol için olduğunu(all, TCP, UDP, ICMP), "kaynak IP uzayı" kısmı kuralın uygulanacağı kaynak IP uzayını, "kaynak maskesi" kısmı kuralın uygulanacağı kaynak IP aralığını, "kaynak port" kısmı kuralın uygulanacağı kaynak portu, "hedef IP uzayı" kısmı kuralın uygulanacağı hedef IP uzayını, "hedef maskesi" kısmı kuralın uygulanacağı hedef IP aralığını, "hedef port" kısmı kuralın uygulanacağı hedef portu, "arayüz" kısmı kuralın uygulanacağı arayüzü, "çift yön" kısmı kuralın her iki yönlüde geçerli olup olmadığını, "ters bağlantı isteği" kısmı

kuralın ters yönlü olarak bağlantı kurma isteği(SYN biti) için olup olmadığını belirtmektedir. Değer bölgesi içinde kullanılmayan alanlar "#" işareti ile doldurulmaktadır. Güvenlik duvarı kuralının giriş, çıkış veya iletme kuralı olduğu jetonlardan anlaşılmaktadır. Giriş için "input?", çıkış için "output?" ve iletme için "forward?" simgesi kullanılmaktadır. "?" sembolü, değeri doğal sayı olan bir değişkeni temsil etmektedir. Tablo 5.2'de güvenlik duvarı için örnek jeton-değer çiftleri görülmektedir.

Tablo 5.2 – Güvenlik duvarı örnek jeton-değer çiftleri

JETON	DEĞER
input0	accept_tcp_122.122.122.0_24_#_0.0.0.0_0_80_eth0_-b_#
input1	deny_tcp_122.122.122.246_32_#_122.122.122.193_32_23_eth0_#_#
output0	deny_icmp_122.122.122.193_24_8_#_#_#_eth0_#_#
forward0	accept_all_-m_122.122.122.0_24_#_0.0.0.0_0_ #_ppp0_# _#

Tablo 5.2'de, "input0" jetonunun değeri :

"ipfwadm -I -a accept -P tcp -S 122.122.122.0/24 -D 0.0.0.0/0 80 -W eth0 –b" komutunun yürütülmesi sonucunda oluşturulmaktadır. Bu komut eth0 arayüzüne gelen 122.122.122.0 ağından bir paketin hedef portu 80 ve hedef adresi ne olursa geçmesine izin verilmesini sağlamaktadır. Aynı şekilde "-b" parametresi ile bu komutun terside geçerli kılnmıştır.

"input1" jetonunun değeri :

"ipfwadm -I -a deny -P tcp -S 122.122.122.246/32 -D 122.122.122.193/32 23 -W eth0" komutunun yürütülmesi sonucunda oluşturulmaktadır. Bu komut eth0 arayüzüne gelen 122.122.122.246 kaynak adresli, 122.122.122.193 hedef adresli ve telnet hedef portlu paketlerin geçişine izin verilmemesini sağlamaktadır.

"output0" jetonunun değeri :

"ipfwadm -O -a deny -P icmp -S 122.122.122.193/32 8 -W eth0" komutunun yürütülmesi sonucunda oluşturulmaktadır. Bu komut eth0 arayüzünden 122.122.122.193 kaynak adresli "icmp Echo Request" paketlerinin çıkmasını engellemektedir.

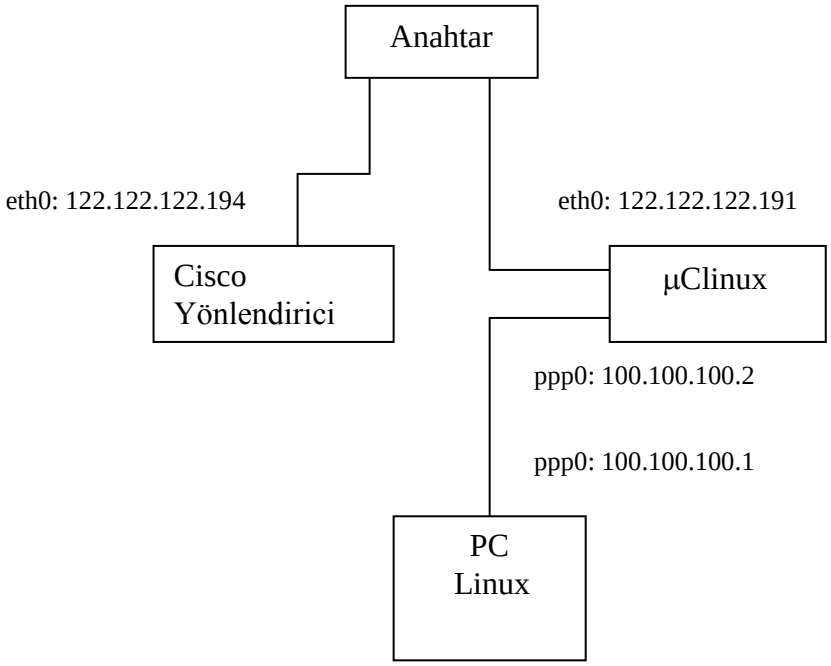
"forward0" jetonunun değeri :

"ipfwadm -F -a accept -m -P all -S 122.122.122.0/24 -D 0.0.0.0/0 -W ppp0" komutunun yürütülmesi sonucunda oluşturulmaktadır. Bu komut 122.122.122.0

ağından gelen paketlerin hedef IP uzayı ne olursa olsun ppp0 arayüzünden maskelenerek yollanmasını sağlamaktadır.

Güvenlik duvarı kurallarının kalıcı belleğe sıralı biçimde yazılması gerekmektedir. Çünkü güvenlik duvarı kuralları kendi kategorisine(input, output, forward) göre ayrı ayrı, sıra ile işlenerek kararlar verilmektedir. Örneğin sistemde input0, input1, input2, input3 değerlerini taşıyan giriş filtreleme ile ilgili jetonlar olsun. Bir kural eklenirken kural tablosunun başına veya sonuna eklenmektedir. Eğer kural tablonun başına eklenecekse kalıcı bellekteki input jetonları birer ötelenmektedir. Yeni kural "input0" jetonuna değer olarak yazılmaktadır. Kural tablosunun sonuna eklenecekse kalıcı bellekteki bütün "input" jetonları taranmakta ve en sona yeni jeton ilave etmektedir. Örneğe göre yeni kural "input5" jetonuna değer olarak yazılmaktadır. Bir kural çıkarılacağı zaman jetonlar çıkarılacak kuralın jeton numarasından itibaren geriye birer numara ötelenmektedirler. Örneğe göre "input1" jetonunun çıkarılacağı düşünülürse, "input2" ve "input3" jetonları geriye doğru birer ötelenmektedir. Yani bu jetonların yeni isimleri sıra ile "input1" ve "input2" olmaktadır.

Sistemin sahip olduğu yönlendirme ve güvenlik duvarı özelliklerini sınamak amaçlı bir test ortamı kurulmuştur. Bu test ortamı Şekil 5.21'de görülmektedir.



Şekil 5.21- Sistem test ortamı

Cisco yönlendiricinin ethernet arayüzü anahtara bağlanmış ve 122.122.122.194 internet adresi atanmıştır. Üzerinde µClinux koşan geliştirme kitinin ethernet arayüzü anahtara bağlanmış ve internet adresi olarak 122.122.122.191 verilmiştir. Kitin birinci seri arabirimi ile üzerinde linux koşan kişisel bilgisayarın sıfırıncı seri arabirimi arasında 115200 b/s hızında PPP bağlantısı kurulmuştur. Geliştirme kitinin

ppp0 arayüzüne 100.100.100.2, kişisel bilgisayarın ppp0 arayüzüne 100.100.100.1 internet adresi verilmiştir.

Cisco yönlendirici üzerinden kişisel bilgisayara "ping" isimli program yardımıyla paketler atılmıştır. Bu paketler kişisel bilgisayar ulaştığı tespit edilmiştir. Aynı işlem kişisel bilgisayardan Cisco yönlendiriciye doğru yapılmıştır. Atılan paketlerin yönlendiriciye ulaştığı tespit edilmiştir. Bu şekilde sistemin yönlendirici işlevi gördüğü ispatlanmıştır.

Sisteme güvenlik duvarı kuralları eklenmiş ve bu kurallara göre Cisco yönlendiricisinden kişisel bilgisayara paketler atılmıştır. Güvenlik duvarı kurallarına göre bu paketlerin filtrelendiği tespit edilmiştir. Örnek olarak, Cisco yölendiriciden kişisel bilgisayara "ICMP" paketleri atılmıştır. Bu paketlerin atılmasında "ping" programı kullanılmıştır. Geçit duvarı üzrinde "ICMP" paketlerinin geçişi engellenmiştir. Dolayısıyla bu paketler ile ilgili cevap alınamamıştır.

Sistemin internet performansını ölçmek için sistem internete bağlanmış ve internet üzerinden 5-6 kilosekizlik bir dosya indirim hızına ulaşılmıştır. Bu hızda telefon bağlantısı üzerinden internete erişimde iyi bir hız teşkil etmektedir.

"ping " programı kullanılarak Şekil 5.21'deki test ortamında, IP paketlerinin Cisco yönlendirici ile PC Linux arasındaki erişim süreleri araştırılmıştır. Cisco yönlendiriciden kişisel bilgisayara ve ters yönde "ping" programı aracılığı ile "ICMP" paketleri atılmıştır. PC Linux ile μClinux arasındaki seri iletişim hızı 57600bps ve 115200bps iken Cisco yönlendiriciden(122.122.122.194) PC Linux'a(100.100.100.1) "ping" programı kullanılarak çeşitli uzunluklarda ve miktarlarda IP paketleri atılmıştır. Tablo 5.3'te elde edilen süreler görülmektedir.

Tablo 5.3 – Cisco yönlendirici→PC Linux paket ulaşım süreleri

Datagram boyutu(sekizli)	Paket sayısı	minimum (milisaniye)	Maksimum (milisaniye)
PC Linux → μClinux seri arabirim hızı: 57600bps			
100	50/250/500	48	52
500	50/250/500	188	192
1000	50/250/500	356	364
2000	50/250/500	716	724
PC Linux → μClinux seri arabirim hızı: 115200bps			
100	50/250/500	28	32
500	50/250/500	100	112
1000	50/250/500	188	196
2000	50/250/500	368	392

Tablo 5.3'de verilen minimum ve maksimum erişim süreleri sürekli değişiklik göstermektedir. Yapılan ölçümler sonucunda elde edilen sonuçları göstermektedir.

μClinux ve PC Linux arasında asenkron seri iletişim olduğundan bu değerlerde belirli sonuçlar yakalanamamıştır. Her defasında datagram boyutları sabit olan paketlerden 50/250/500 adet yollanmıştır. Sonuçlar değişiklik gösterdiğinden dolayı karşılaşılan en düşük değer ve en yüksek değerler Tablo 5.3'e yazılmıştır. PC Linux ile μClinux arasındaki seri iletişim hızı 57600bps ve 115200bps iken PC Linux'dan(100.100.100.1) Cisco yönlendiriciye(122.122.122.194) "ping" programı kullanılarak çeşitli uzunluklarda ve miktarlarda IP paketleri atılmıştır. Tablo 5.4'te elde edilen süreler görülmektedir.

Tablo 5.4 – PC Linux → Cisco yönlendirici paket ulaşım süreleri

Datagram boyutu(sekizli)	Paket sayısı	minimum (milisaniye)	Maksimum (milisaniye)
PC Linux → μClinux seri arabirim hızı: 57600bps			
100	50/250/500	49.600	49.950
500	50/250/500	189.930	189.960
1000	50/250/500	369.925	369.942
2000	50/250/500	719.930	719.942
PC Linux → μClinux seri arabirim hızı: 115200bps			
100	50/250/500	29.930	29.941
500	50/250/500	99.949	99.945
1000	50/250/500	189.934	189.949
2000	50/250/500	369.920	369.935

Güvenlik duvarı kurallarının sistem performansına etkileri amaçlı olarak, sistem güvenlik duvarı yönetimini yapan "ipfwadm" programına eklentiler yapılmıştır. Bu program sayesinde kaynak adresi, kaynak portu, hedef adresi, hedef portu ve protokol cinsi verilmiş bir paketin sistem güvenlik duvarından geçip geçmeyeceği anlaşılmaktadır. Bu özellik, pakete ilişkin parametrelerin "ipfwadm" programı tarafından çekirdeğe aktarılması ve çekirdeğin ilişkisel güvenlik duvarı tablolarına(giriş, çıkış, iletme) bakarak uygun cevap(accep, deny, reject, masquerading) döndürmesiyle sağlanmaktadır. Kaynak adresi 100.100.100.0, kaynak maskesi 255.255.255.0, kaynak portu 56, hedef adresi 200.200.200.0, hedef maskesi 255.255.255.0, hedef portu 57 olan bir paket için giriş güvenlik duvarı tablosuna 1.sıradan 10.sıraya kadar bir kural yerleştirilmiştir. Her bir yerleştirimden sonra "ipfwadm –I –c –P tcp 100.100.100.0/24 56 –D 200.200.200.0/24 57 –V 122.122.122.190 –W eth0" komutu çalıştırılarak herbir kuralın sisteme getirdiği ek süre hesaplanmıştır. Sürenin daha kesin ölçülebilmesi için "ipfwadm" programı içinde kullanılan "for" döngüsü 10.000.000'a kadar devam etmektedir. Her bir adımda çekirdeğe paket parametreleri aktarılmaktadır. Döngüye girmeden önce sistem saati ile zaman ölçülmekte, döngü çıkışında da sistem zamanı ölçülmektedir. İki zaman arasındaki fark 10.000.000'a bölünerek çekirdeğin paket kontrolünde harcadığı zaman hesaplanmaktadır. Bu hesaplama kontrol edilen paket için 1. sıradan

10.sıraya kadar tekrarlanmaktadır. Ölçülen değerler saniye cinsindendir. Temel ölçüm hatası 1/10.000.000'dur. Bu da 0,1 μs etmektedir. Çekirdeğin paket kontrolünde harcadığı süreler Tablo 5.5'de gösterilmektedir. Tablo 5.5'deki süre değerleri saniye cinsinden ölçüm değerlerinin 10.000.000'a bölünmesiyle bulunmuştur.

Tablo 5.5– Paket kontrol süreleri

Kural sırası	Süre(μs)
Tablo boş	99
1	109,5
2	116,1
3	120,6
4	127
5	131,9
6	137,9
7	143
8	148,6
9	154,2
10	159,9

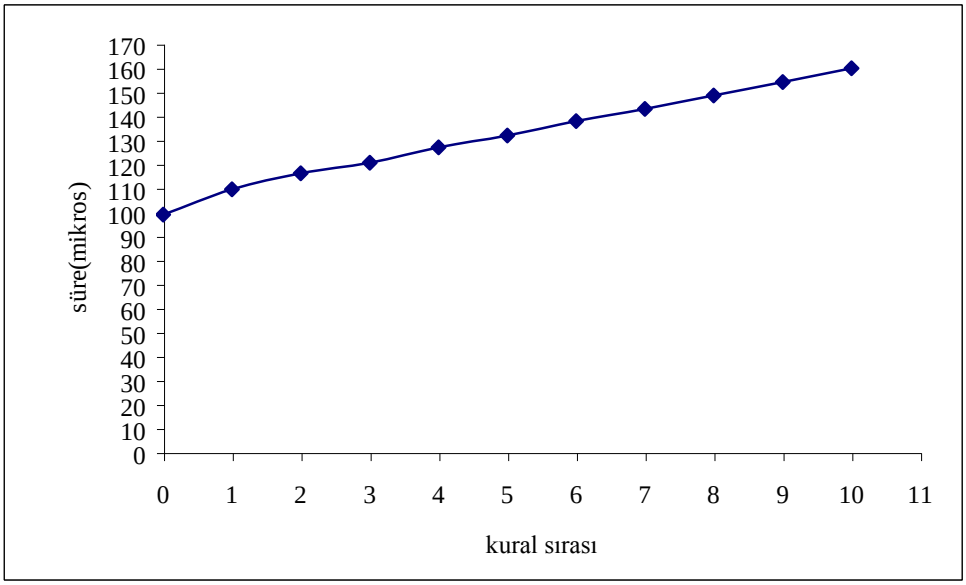
Tablo5.5'deki değerlere göre her bir kuralın ortalama işlenme süresi şu şekilde hesaplanmıştır:

159,9-109,5=50,4μs.

Ortalama kural işlenme süresi=50,4/9=5,6μs.

Süre hesabına güvenlik duvarı tablosu boşken elde edilen sonuç dikkate alınmamıştır. Bunun nedeni çekirdeğin güvenlik duvarı tablosunda kural yokken ve kural varken yaptığı işlemlerin farklı olmasıdır. Çekirdek güvenlik duvarı tablosunda kural varken "/proc" dizini altındaki ilişkisel dosyayı açmaktadır. Dolayısıyla dosya açma ve kapama işlemi ek süre getirmektedir.

Güvenlik duvarı kurallarının sisteme getirdiği ek süre doğrusal olarak artmaktadır. Bir foksiyon şeklinde ifade etmek gerekirse f(x)= ax+b şeklindedir. "x" değerleri kural sayısını, b değeri giriş tablosunda hiç kural yokken çekirdeğin kural kontrolü için yaptığı işlemlerin süresini belirtmektedir. Bu değerde 99 μs'dir. "a" katsayısı her bir kuralın işlenme süresini belirtmektedir. Şekil 5.22'deki grafikten de görüldüğü üzere "a" katsayısı sabit bir süreyi göstermektedir. Değeride yaklaşık olarak 5,6μs'dir.



Şekil 5.22- Güvenlik duvarı kural sırası-süre grafiği

Tablo 5.6'da performans incelemesi için sistemin giriş güvenlik tablosuna eklenilen kurallar sıra ile görülmektedir.

Tablo 5.6 - Güvenlik duvarı kuralları

Kaynak Adres	Kaynak Maskesi	Kaynak Port	Hedef Adresi	Hedef Maskesi	Hedef Port
100.100.100.0	24	56	200.200.200.0	24	66
100.100.100.0	24	56	200.200.200.0	24	65
100.100.100.0	24	56	200.200.200.0	24	64
100.100.100.0	24	56	200.200.200.0	24	63
100.100.100.0	24	56	200.200.200.0	24	62
100.100.100.0	24	56	200.200.200.0	24	61
100.100.100.0	24	56	200.200.200.0	24	60
100.100.100.0	24	56	200.200.200.0	24	59
100.100.100.0	24	56	200.200.200.0	24	58
100.100.100.0	24	56	200.200.200.0	24	57

6. SONUÇ

Ağ uygulamalarında kullanılmak amaçlı geliştirilen gömülü sistemlerde, işletim sistemi olarak μClinux'un kullanılması başta fiyat olmak üzere birçok avantaj sağlamıştır. Geliştirme kiti dışında işletim sistemi kodları ve uygulamalar için para ödenmemiştir. Yazılım geliştirme ortamı yeterlidir. Ancak bu ortamı kullanabilmek için uğraştırıcı bir öğrenme evresinden geçilmektedir. Yazılım geliştirirken ve işletim sistemini istenilen amaca yönelik özelleştirilirken çıkan problemlerin çözülmesinde belirli bir teknik destek ekibinin olmaması yapılan projenin ilerleyişini aksatmıştır. Problemlerin çözümünde elektronik posta gruplarından faydalanılmıştır. Sistemin, IP-paylaştırıcı IP-güvenlik duvarı, IP-yönlendirici işlevlerini gerçekleyip gerçeklemediğine dair bir test ortamı kurulmuş ve bu işlevleri gerçeklediği anlaşılmıştır. Sistemin Internet üzerinden 5-6 kilosekizlik bir dosya indirme hızına imkan vermiştir. Bu da telefon bağlantısı için çok iyi bir hız teşkil etmektedir. Güvenlik duvarı kurallarının sisteme getirdiği yük incelenmiştir. Her bir kuralın sisteme getirdiği yük ortalama 5,6μs'dir. Kural eklenmesi sisteme doğrusal artan bir yük getirmiştir. Yazılımların bir kısmı Linux üzerinde geliştirilmiş ve hiçbir ek çaba gerekmeden μClinux ortamına uyarlanmıştır. Dolayısıyla bu durum yazılım geliştirmede platform bağımsız bir ortam sunmuştur. Diğer gömülü işletim sistemleri üzerinde çalışma imkanı bulunmamasına rağmen, bu işletim sistemleri üzerinde deneyler yapan ve raporlar hazırlayan internet sitelerinden faydalanılmıştır. Üretilen sistemin amacına göre seçilecek işletim sistemine karar vermek gerekmektedir. Proje için konulan para ve amaç orta ölçekli ise yaygın olarak kullanılması, teknik desteğinin çok iyi olması, pazarlamayı iyi yapmaları, referanslarının iyi olması ve elde edilen değerlendirme raporları sonucunda VxWorks işletim sisteminin tercih edilmesi sonucuna varılmıştır. Ancak μClinux her geçen gün ileriye yönelik adım atmakta ve gömülü işletim sistemi dünyasında pazar payını arttırmaktadır. Dolayısıyla μClinux'un gömülü işletim sistemi olarak seçilmesi doğru olacaktır.

EKLER

Ek A

A.1 İş Yönetimi

Tablo A.1- İş yönetimi

	QNX 6.1	Vxworks 5.3.1	Windows CE 3.0
Model	Süreçler ve görevcikler	Her iş, tek görevcik yürütümü içermekte. Bütün işler, hiçbir koruma olmadan aynı adres uzayında çalışmaktadır. VxVMI ile her işin kendi çalışma uzayı olmaktadır.	Süreçler ve görevcikler
Öncelik Seviyesi	64	256	256
Maksimum İş Sayısı	4095 süreç. Her süreç 32767 görevciğe sahip olabilir.	Bellek boyutuna göre sınırlandırılmıştır.	32 süreç. Bir sürecin sahip olabileceği görevcik miktarı sahip olunan bellek boyutuna bağlıdır.
Programlama Metodları	Öncelikleştirilmiş FIFO. Döngü programlama.	POSIX ve "Wind" programlama Döngü programlama.	Döngü (düzenlenebilir zaman aralıkları ile).
Durum Sayısı	14	9	5(çalışma, erteleme, uyuma, bloke, sonlandırılmış)

A.2 Bellek Yönetimi

Tablo A.2- Bellek yönetimi

	QNX 6.1	Vxworks 5.3.1	Windows CE 3.0
MMU Desteği	Var	Var(VxVMI ile)	Var
Fiziksel Sayfa Büyüklüğü	64	8K(öndeğer)	1K veya 4K (sisteme bağlı olarak)
Yer değiştirme/Talep Sayfalama	Var/Yok	Yok/Yok	Var/Var(gerçek zaman performansı için kapatılabilir.)
Sanal Bellek	Var	Var(VxVMI ile)	Var
Bellek Koruma Modeli	Tam sanal bellek koruma	Koruma yok. VxVMI ile her süreç kendi bellek alanında çalışabiliyor.	Tam sanal bellek koruma

A.3 Kesme Kotarımı

Tablo A.3- Kesme kotarımı

	QNX 6.1	VxWorks 5.3.1	Windows CE 3.0
Kotarım	İç içe, öncelikli	İç içe, öncelikli	İç içe, öncelikli
İçerik	ISR, kendisine tutturulan görevcik bağlamında çalışır.	Kesme kotarıcıları, diğer işlerin bağlamı dışında özel bir bağlamda çalışır.	ISR, özel bir bağlamda çalışmakta ve OEM tarafından belirlenmiş sanl bellek adreslerini kullanmaktadır. ISG, kendi bağlamında çalışan görevcik.
Yığın	ISR, kendi yığınına sahiptir.	Özel kesme yığınında çalışır.Eğer donanım buna izin vermiyorsa kesilen işin yığını kullanılır.	ISG, kendine ait bir yığını vardır.
Kesme-iş iletişimi	Sinyaller ve darbeler	Paylaşılan bellek ve halka arabellek. Semaforlar (sadece serbest bırakma) Mesaj kuyrukları (sadece yollama) Boru (sadece yazma) Sinyaller (sadece yollama)	ISR içinde, ISG'ye işaret etmek için olay kullanılır.
Minimum RAM	-	100 sekizliden az.	-

A.4 API zenginliği

Tablo A.4a- API zenginliği

İş yönetimi	QNX 6.1	Vxworks 5.3.1	Windows CE 3.0
Yığın boyutunu almak	Var	Var	Var
Yığın boyutunu kurmak	Var	Var	Var
Yığın adresini almak	Var	Var	Var
Yığın adresini kurmak	Var	Var	-
Görevcik durumunu almak	Var	Var	Var
Görevcik durumunu kurmak	Var	Var	Var
TCB'yi almak	Var	Var	Var
TCB'yi kurmak	-	Var	Var
Önceliği bilgisini almak	Var	Var	Var
Önceliği ayarlamak	Var	Var	Var
Görevcik belirtecini almak	Var	Var	Var
Görevcik durum değiştirme kotarıcısı	-	Var	-
Aktif yığın işaretçisini almak	Var	Var	Var
Görevcik CPU kullanımını kurmak	Var	-	-
Programlama mekanizmasını ayarlamak	Var	Var	Var
Görevciği bellekte kilitlemek	Var	-	-
Programlamayı etkinsizleştirmek	-	Var	-
Toplam	14	15	12

Tablo A.4b – Sabit uzunluklu bellek parçası için API'lar

Sabit blok uzunluklu parça	QNX 6.1	Vxworks 5.3.1	Windows CE 3.0
Parça boyutunu ayarlamak	-	Var	-
Parça boyutunu almak	-	Var	-
Bellek blok boyutunu ayarlamak	-	Var	-
Bellek blok boyutunu almak	-	Var	-
Parça yerini belirlemek	-	-	-
Bellek bloğunu almak-bloke etmek	-	-	-
Bellek bloğunu almak-bloke etmeden	-	Var	-
Bellek bloğunu almak-zaman aralığı ile	-	-	-
Bellek bloklarını serbest bırakmak	-	Var	-
Parçaları uzatmak	-	Var	-
Serbest bellek bloklarının sayılarını almak	-	Var	-
Bellekteki parçaları kilitlemek/açmak	-	-	-
Toplam	-	8	-

Tablo A.4c – Sabit uzunluklu olmayan blok havuzu için API'lar

Sabit olmayan blok havuzu	QNX 6.1	Vxworks 5.3.1	Windows CE 3.0
Havuz boyutunu almak	Var	Var	Var
Havuz boyutunu ayarlamak	Var	Var	-
Yeni bir havuz yaratmak	-	Var	Var
Bellek blok boyutunu almak	-	Var	Var
Bellek bloğunu alma-bloke etmek	Var	-	-
Bellek bloğunu almak-bloke etmeden	-	Var	Var
Bellek bloğunu almak-zaman aralığı ile	-	-	-
Bellek bloklarını serbest bırakmak	Var	Var	Var
Havuzu genişletmek	-	Var	Var
Blokları genişletmek	Var	Var	Var
Geriye kalan serbest baytları almak	-	Var	-
Bellekteki havuzu kilitlemek/açmak	Var	-	Var
Bellekteki blokları kilitlemek/açmak	Var	-	Var
Toplam	7	9	9

A.5 Araçlar

Tablo A.5 - Araçlar

	QNX Vxworks CE	IDE içinde	Tek başına	Komut tabanlı	GUI tabanlı
Editör					
Renkli gösterilim	Var Var Var	Evet Evet Evet	- - -	- - -	Evet Evet Evet
Yardım	Var Var Var	Evet Evet Evet	- - -	- - -	Evet Evet Evet
Otomatik kod gösterilimi	Var Yok Yok	Evet - -	- - -	- - -	Evet - -
Derleyici					
C	Var Var Var	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet
C ++	Var Var Var	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet
Ada	Yok Yok Yok	- - -	- - -	- - -	- - -
Assembler	Var Var Var	Evet - Evet	Evet Evet Evet	Evet Evet Evet	Evet - Evet
Bağlayıcı					
Artan	Var Var Var	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet
Sembol tablosu oluşturma	Var Var Var	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet	Evet Evet Evet
Yükleyici					
TFTP başlangıç yükleyicisi	Var Var Var	- Evet Evet	- - Evet	- - -	 Evet Evet
Seri başlangıç yükleyicisi	Var Var Var	- Evet Evet	- - Evet	- - -	 Evet Evet
Modülleri ayrı yüklemek	Yok Var Yok	- Evet -	- - -	- - -	- Evet -

KAYNAKLAR

[1] http://www-2.cs.cmu.edu/afs/cs/usr/wing/www/mit/leveson/node1.html

[2] http://www.netrino.com/Publications/Glossary

[3] http://www.y-axis.com/technologythisweek

[4] http://www.emlinux.com/paper.html

[5] http://sepc.twi.tudelft.nl/~toet/intra/html/html/es/node2.html

[6] http://www.li.org/linuxhistory.php

[7] http://www.uClinux.org

[8] http://www.dedicated-systems.com/magazine/97q2/winntasrtos.htm

[9] http://www.mitre.org/support/swee/html/65_hutto/sld012.htm

[10] http://www.dedicated-systems.com/

[11] http://www.qnx.com

[12] http://www.vxworks.com

[13] **Tanenbaum, A.S ve Woodhull A.S**, 1997. Operating Systems:Design and implementation. Prentice-Hall Inc., New Jersey

[14] http://www.windriver.com

[15] http://www.microsoft.com/windows/embedded

[16] http://www.linuxdevices.com/articles/AT9848697606.html

[17] http://www.fsmlabs.com

[18] **Ungerer G.,**2002, E-mail görüşmesi

[19] http://e-www.motorola.com/brdata/PDFDB/docs/AN2005.pdf

[20] www.motorola.com

[21] **Mancill T**., 2001, Linux Routers, Prentice-Hall Inc., New Jersey.

[22] **Boutell T**., 1996, CGI Programming in C&Perl, Addison-Wesley., California.

[23] **Kernighan B.W., Ritchie D. M.,** 1988, The C Programming Language., Prentice-Hall Inc., New Jersey.