MEHMET AKKİF ERSOY UNİVERSİTESİ

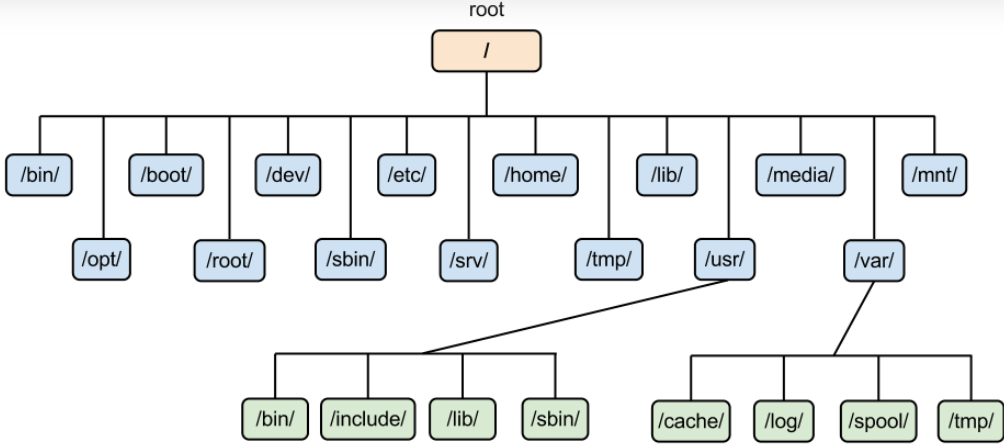
İŞLETİM SİSTEMLERİ VİZE ÖDEVİ

BATUHAN BAYKAL 1611404006

1)**Linux Dosya Yönetim Sistemi**

Bir *dosya sistemi* bir işletim sistemi kullanan bir disk veya disk üzerinde dosyaların izini sürmek için yöntemler ve veri yapıları olduğu; diğer bir deyişle, dosyaların diskte düzenlenme şekli. Sözcük ayrıca, dosyaları veya dosya sisteminin türünü depolamak için kullanılan bir bölüm veya diski ifade etmek için kullanılır. Bu nedenle, biri iki dosya sistemine sahip olduğum, birinin dosyaları depoladığı iki bölüme sahip olduğu veya dosya sisteminin türü anlamına gelen genişletilmiş dosya sistemini kullandığı söylenebilir.

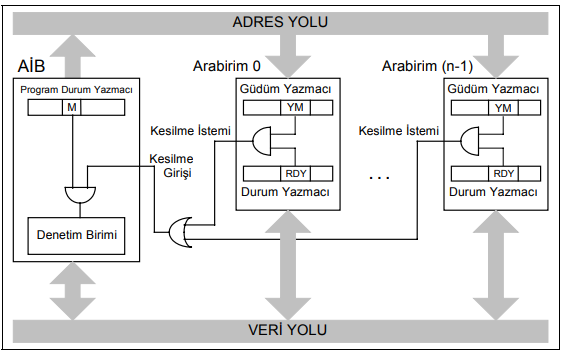
Çoğu UNIX dosya sistemi türü benzer bir genel yapıya sahiptir, ancak kesin ayrıntılar biraz farklılık gösterir. Ana kavramlar *süper blok* , *inode* , *veri bloğu* , *dizin bloğu* ve *dolaylı bloktur*. Süper blok, boyutu gibi bir bütün olarak dosya sistemi hakkında bilgi içerir (buradaki kesin bilgi dosya sistemine bağlıdır). Bir inode, adı dışında bir dosya hakkındaki tüm bilgileri içerir. Ad, inode numarasıyla birlikte dizinde saklanır. Bir dizin girişi, bir dosya adı ve dosyayı temsil eden inode numarasından oluşur. İnode, verileri dosyada saklamak için kullanılan birkaç veri bloğunun numaralarını içerir. Ancak, inode'da yalnızca birkaç veri bloğu numarası için yer vardır ve daha fazlasına ihtiyaç duyulursa, veri bloklarına işaretçiler için daha fazla alan dinamik olarak tahsis edilir. Bu dinamik olarak tahsis edilmiş bloklar dolaylı bloklardır; isim, veri bloğunu bulmak için, önce dolaylı bloktaki numarasının bulunması gerektiğini belirtir.

UNIX gibi, Linux da tek bir hiyerarşik dizin yapısına sahip olmayı seçer. Her şey / ile temsil edilen kök dizinden başlar ve ardından "sürücüler" denen şey yerine alt dizinlere genişler. Windows ortamında, kişi dosyalarını hemen hemen her yere koyabilir: C sürücüsüne, D sürücüsüne, E sürücüsüne, vb. Böyle bir dosya sistemine hiyerarşik yapı denir ve işletim tarafından değil, programların kendileri (program dizinleri) tarafından yönetilir. sistemi. Öte yandan, Linux, önyükleme sürecindeki önemlerine göre dizinleri / kök dizininden azalan şekilde sıralar.

Linux'un neden ters eğik çizgi \ yerine frontslash \ kullandığını merak ediyorsanız, bunun nedeni sadece UNIX geleneğini takip ediyor olmasıdır. Unix gibi Linux da büyük / küçük harfe duyarlı olmayı seçer. Bunun anlamı, büyük harf olsun ya da olmasın karakterlerin durumunun çok önemli hale gelmesidir. Yani bu, BU ile aynı değil. Bu özellik, özellikle dosya aktarım işlemleri sırasında, ister disket gibi taşınabilir disk ortamıyla isterse FTP yoluyla kablo üzerinden olsun, yeni kullanıcılar için oldukça büyük bir sorun oranını açıklamaktadır.

Bir bölüm veya aygıt bağlamadıkça, sistem bu bölümün veya aygıtın varlığından haberdar olmaz. Bu, bölümlerinize veya aygıtlarınıza erişim sağlamanın en kolay yolu gibi görünmeyebilir, ancak diğer işletim sistemlerine kıyasla çok daha fazla esneklik avantajı sunar. Birleşik dosya sistemi olarak bilinen bu tür bir düzen, Windows'un kullandığı yaklaşıma göre çeşitli avantajlar sunar. / Usr dizini örneğini ele alalım. Kök dizinin bu alt dizini, sistem çalıştırılabilirlerinin çoğunu içerir. Linux dosya sistemiyle, NFS (Sun), Coda (CMU) veya AFS (IBM) gibi sayısız protokol kümesi kullanarak onu başka bir bölüme veya hatta başka bir makineye bağlamayı seçebilirsiniz. Temeldeki sistem farkı bilmeyecek ve bilmesine gerek olmayacaktır. / Usr dizininin varlığı tamamen şeffaftır. Yerel dizin yapısının parçası olan yerel bir dizin gibi görünüyor.

2)



3 adet arabirimin yer aldığı sistemin register ve adres değerleri verilen özelliklere göre hesaplanmıştır. Buna göre;

**Arabirim 0’ın özellikleri:**

İletim hızı: 9600bps

Aktarılan veri bit sayısı: 8

Bitiş bit uzunluğu: 1

The parity bit: Çift

RE: 1

TE: 1

Taşma biti hatası: hatasız

Eşlik biti hatası: hatasız

RR: hazır

TR: hazır

**Arabirim 1’ın özellikleri:**

İletim hızı: 4800bps

Aktarılan veri bit sayısı: 8

Bitiş bit uzunluğu: 1

The parity bit: Çift

RE: 1

TE: 1

Taşma biti hatası: hatasız

Eşlik biti hatası: hatasız

RR: hazır

TR: hazır

**Arabirim 2’ın özellikleri:**

İletim hızı: 2400bps

Aktarılan veri bit sayısı: 16

Bitiş bit uzunluğu: 1

The parity bit: Çift

RE: 1

TE: 1

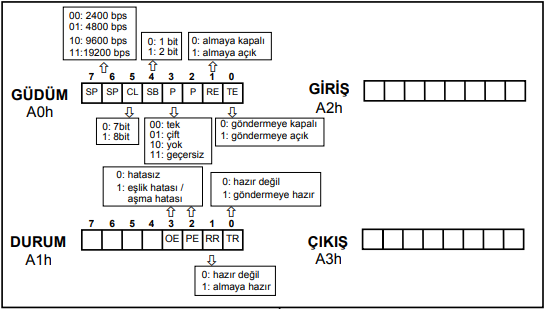
Taşma biti hatası: hatasız

Eşlik biti hatası: hatasız

RR: hazır

TR: hazır

Arabirimlerin yer aldığı bilgisayar sistemi giriş/çıkış adres evreninde; güdüm yazmacının A0H, durum yazmacının A1H fiziksel adresine atandığı varsayılmaktadır. O halde, bir önceki sayfada yer alan görselde yer alan arabirimimizin A1H ve A0H adres değerlerini verilen sistem özelliklerine göre hesaplayalım. Sistem özelliklerine göre oluşan register içeriği aşağıda yer alan görseldeki protokole göre sıralanır.



Yukarıdaki protokolden ve arabirimin özelliklerinden yola çıkarak, arabirim 0’ın A0H(güdüm) ve A1H(durum) içeriği;

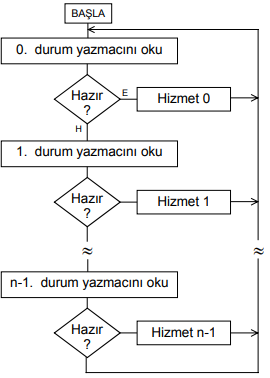
A0H(10100111) ve A1H(0011) şeklindedir.

Yukarıdaki protokolden ve arabirimin özelliklerinden yola çıkarak, arabirim 1’ın A0H(güdüm) ve A1H(durum) içeriği;

A0H(01100111) ve A1H(0011) şeklindedir.

Yukarıdaki protokolden ve arabirimin özelliklerinden yola çıkarak, arabirim 1’ın A0H(güdüm) ve A1H(durum) içeriği;

A0H(00000111) ve A1H(0011) şeklindedir.



Birden çok giriş/ çıkış biriminin bulunduğu bir sistemde, bunlardan iki ya da daha çoğunun aynı anda (veri yastıklarının okunup yazılması gibi) hizmet istemleri söz konusu olduğunda, hangi birime daha önce hizmet verileceği sorunu ortaya çıkar. Bu sorun öncelik sorunu olarak bilinir. Seçmeli giriş/ çıkış programlama yönteminde birimler arası öncelikler arabirim durum yazmaçlarının taranma sıra ve sıklığıyla belirlenir. En öncelikli birim en sık taranan birimdir. Eğer bir sistemde, 0 'dan (n-1)'e kadar numaralanan n adet giriş/ çıkış birimi bulunduğu düşünülürse, en öncelikli olduğu varsayılan 0’ıncı birimin durum yazmacı öncelikle okunur, ilgili durum bitleri aracılığıyla birimin hizmet gereksinimi bulunup bulunmadığı sınanır. Eğer birim hizmet gerektiriyorsa ilgili yordama sapılır. Yordam dönüşünde yine bu birimden başlamak üzere tarama işlemi yinelenir. Eğer birim hizmet gerektirmiyorsa bir sonraki birimin 48 İŞLETİM SİSTEMLERİ durum sınamasına geçilir. i inci birimin hizmet almak için hazır olmadığı durumda (i+1)inci birimin durum sınamasına geçilir. i inci birimin durum sınaması sonucu hizmet yordamına sapılmışsa bu yordamdan geri dönüşte sınamalara 0’ıncı birimden, yeniden başlanır. Sistemde yer alan tüm birimlerin eş öncelikli oldukları varsayıldığında taramalar, Çizim 2.10.b'deki akış çizgesine göre yapılır. Bu iki yaklaşım arasındaki tek ve önemli ayrım, hizmet yordamlarından geri dönüşte başa (0’ıncı birime) ya da izleyen (i+1’inci) birimin sınanmasına geçilmesidir.

**KAYNAKLAR**

1) <https://www.tutorialspoint.com/unix/unix-file-management.htm>

2) <https://www.rs-online.com/designspark/an-intro-to-linux-file-system-management>

3) https://uzak.mehmetakif.edu.tr/pluginfile.php/317945/mod\_resource/content/1/ALI\_SAATCI.pdf