

```
/*
 * Copyright (c) 2004 Baris Simsek.
 *
 * Permission is granted to copy, distribute and/or modify this document
 * under the terms of the GNU Free Documentation License, Version 1.2
 * or any later version published by the Free Software Foundation;
 * with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
 * Texts. A copy of the license is included in the section entitled "GNU
 * Free Documentation License".
 */
```

## Belge Hakkında

Bu belge 18 Ağustos 2004'te Acikkod.ORG , Açık Kod Yazılım Destekleme Organizasyonu için yazılmıştır.

## Kavramlar

Program kütüphaneleri, daha sonra yeni yazılacak programlara dahil edilmek üzere saklanan derlenmiş hazır kod ve veri dosyalarıdır. Kütüphane dosyaları, modüler programlama imkanı sunarlar.

## Statik Kütüphaneler

Statik kütüphaneler, yeniden derlenmeye ihtiyaç duymaksızın programa derleme aşamasında yani çalıştırılmadan önce eklenir. lib dizinlerindeki “.a” uzantılı dosyalar statik kütüphanelerdir. Derleme zamanını azaltmak için statik kütüphaneler kullanılıyordu. Ancak günümüzün hızlı bilgisayarları için derleme zamanı problem olmaktan çıktı. Bu nedenle statik kütüphaneler artık tercih edilmemektedir. Ancak yine de kütüphane kodlarını açmak istemeyenler için kullanışlıdır.

Statik kütüphane oluşturmak için `ar (1)` komutu kullanılır.

```
# ar rc libtest.a test.o util.o
```

Bu komut “libtest.a” kütüphanesini oluşturur, test.o ve util.o object dosyalarını bu kütüphaneye ekler. “c” parametresi, eğer kütüphane yoksa oluşturulması gerektiğini söyler. Eğer varsa object dosyaları bu kütüphaneye eklenir. “r” parametresi ile eski object dosyaların yenileri ile değiştirilmesi talep edilir.

Yeni oluşturulan veya değiştirilen bir kütüphane dosyasını yeniden indekslemek gerekir. Bu işlem derleme sırasında derleyicinin sembolleri kontrolünü hızlandırmak içindir. İndeks oluşturmak için `ranlib(1)` kullanılır.

```
# ranlib libtest.a
```

Derlenecek bir programa daha önce oluşturulan bir kütüphaneyi dahil etmek için `gcc (1)` ‘ye `-l` ile kütüphane ismi parametre olarak verilmelidir. Bu işlem için tercih edilmese de doğrudan `ld(1)` kullanılabilir.

```
# gcc prog.o -L/home/simsek/libtest -ltest
```

-l ile kütüphane ismini verirken dosya isminin başındaki “lib” in kaldırıldığına dikkat edilmeli. -L parametresi gcc'ye kütüphaneleri ararken hangi dizinleri kullanacağını bildirir.

Aşağıda bir kütüphane oluşturma ve kullanma senaryosu mevcut:

#### **main.c**

```
#include <stdio.h>

/* fonksiyonlar external tanımlanmalı */
extern void test();
extern void util();

int main() {
    printf("main icinden");
    /* Kutuphane icinden fonksiyon cagiralim */
    test();
    util();

    return 0;
}
```

#### **test.c**

```
int test() {
    printf("Test icerisinden");

    return 0;
}
```

#### **util.c**

```
int util() {
    printf("Util icerisinden");

    return 0;
}
```

```
# gcc -c test.c
# gcc -c util.c
# ar rc libtest.a test.o util.o
# ranlib libtest.a
# gcc -c main.c
# gcc main.o -L /home/simsek -ltest -o testprog
```

Son satırda kütüphane fonksiyonlarını kullanan program derlenip ‘testprog’ isimli çalışabilir dosya derleyici tarafından oluşturuldu. Program çalıştırıldığında aşağıdaki çıktıyı verecektir:

```
# ./testprog
main icinde
Test icerisinde
Util icerisinde
```

## Ortak Paylaşılan (Shared) Kütüphaneler

Paylaşılan (shared) kütüphaneler, program çalıştırıldığında programa dahil edilirler. Programın içerisine gerçekten kod eklenmez. Paylaşımli kütüphaneden bir fonksiyon çağrılan yerlere, kütüphaneye bir referans verilir. Kütüphaneyi kullanan bütün programlar bu şekilde aynı fonksiyon kütüphanesini ortak kullanırlar.

Paylaşımli her kütüphane “soname” denilen özel isme sahiptir. “soname” isimleri gerçek isimlerine bir linktir. Ayrıca derleyicinin bir kütüphaneyi talep ettiğinde kullandığı linker ismi vardır. Bu, versiyonu olmayan bir soname olarak düşünülebilir.

```
$ ls /usr/lib/libz*
libz.a libz.so.1@ libz.so.1.2.1.1*
```

Listede görülen libz.a statik kütüphaneye aittir. libz.so.1 ise libz.so.1.2.1.1 paylaşımli kütüphanesine link olan soname'dir. Bu kütüphane için linker ismi libz.so'dur.

Programlar içerisinde soname'ler tutulur. Bu sayede kütüphanede bir değişiklik yapıldığında yeni kütüphane kullanıma alınmış olunur.

```
$ ldd /bin/ls
librt.so.1 => /lib/librt.so.1 (0x4001a000)
libc.so.6 => /lib/libc.so.6 (0x4002c000)
libpthread.so.0 => /lib/libpthread.so.0 (0x4015b000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

ldd(1) komutu, binary bir programın bağlı olduğu bir başka ifade ile ihtiyaç duyduğu kütüphaneleri tespit eder. Yukarıda 'ls' dizin listeleme komutunun ihtiyaç duyduğu kütüphaneler görülmektedir. Bu listede görülen isimler soname isimleridir.

Ldconfig(8), mevcut kütüphane dosyalarını tespit eder ve soname'leri /etc/ld.so.cache isimli dosyada saklar. ld.so.cache dosyasını oluşturmak için /lib, /usr/lib ve /etc/ld.so.conf dosyasında belirtilen dizinlere bakar. Yeni bir kütüphane eklendiğinde ldconfig çalıştırılmalıdır.

Ortak paylaşımli kütüphane oluşturmak için gcc(1) 'ye '-shared' ile birlikte '-fPIC' veya '-fpic' parametresini vermek gerekir. Paylaşımli kütüphane içindeki bir fonksiyon farklı programlar tarafından kullanıldığı zaman her programda farklı adreslere yerleşirler. Bu nedenle position independent code (PIC) yani adresten bağımsız kod oluşturulması gerekir.

```
$ gcc -c -fpic test.c
$ gcc -shared -o libtest.so test.o
$ ls -l libtest.so
-rwxr-xr-x 1 simsek users 6077 2004-08-17 23:04 libtest.so*
$ ldconfig
```

Daha önce oluşturulan `test.c` kodunda `test()` isimli fonksiyon vardı. Yukardaki işlemler ile bu kod, paylaşımlı kütüphane haline getirildi. Bu kütüphaneyi kullanan bir örnek aşağıda mevcut:

### *shared.c*

```
#include <stdio.h>

/* fonksiyonlar external tanımlanmalı */
extern void test();

int main() {
    printf("main icinden");
    /* Kutuphane icinden fonksiyon cagiralim */
    test();
    return 0;
}
```

```
# gcc -c shared.c
# gcc -o sharedprog shared.o -L. -ltest
# ./sharedprog

main icinde
Test icerisinde
```

“Test icerisinde” mesajı, paylaşımlı kütüphane tarafından sunulan fonksiyon tarafından üretilmektedir.

Derleme aşamasında linker, kütüphaneleri yalnızca `/lib` ve `/usr/lib` altında arar. Eğer oluşturulan kütüphane farklı bir dizinde ise kütüphaneyi bulamayacaktır. Bu problem gcc ile aşılabılır. Gcc'ye, `'-Wl,option'` biçiminde bir parametre verilirse `'options'` olarak verilen seçeneği aynen linker'a iletir. Bu özellik kullanılarak linker'a `'-rpath'` seçeneği ile yeni oluşturulan kütüphanenin dizini bildirilebilir.

```
# gcc -o sharedprog shared.o -L. -ltest -Wl,-rpath,/home/simsek/libtest
```

Bir diğer çözüm ise program çalışırken tıpkı `PATH` kabuk değişkeni gibi `LD_LIBRARY_PATH` kabuk değişkeni tanımlamaktır.

```
# export LD_LIBRARY_PATH=/lib:/usr/lib:/home/simsek/libtest
```

Baris Simsek  
EnderUNIX Software Development Team  
<http://www.enderunix.org/simsek/>