2025-2026 GÜZ YARIYILI İŞLETİM SİSTEMLERİ DERSİ PROJE RAPORU

GRUP 44

Akademisyen: Prof. Dr. Ahmet Zengin

Öğrenci İsimleri/Numaraları/Şubeleri:

Petek İrem Hızlı- G231210030 – 1A

Ramazan Bıyık- G231210088 - 2B

Zeynep Azap- G231210056 -2C

Ömer Akkıyal- G231210058 -1B

Ramazan Katran- G231210076 -2B

GitHub Linki: https://github.com/Ramazanbiyiik/isletim_sistemleri_projesi

21 ARALIK 2025

1. Giriş: FreeRTOS ve Gerçek Zamanlı İşletim Sistemleri

Gerçek zamanlı işletim sistemleri (Real-Time Operating Systems – RTOS), görevlerin yalnızca doğru sonuç üretmesini değil, bu sonuçları belirli zaman kısıtları içinde üretmesini garanti eden sistemlerdir. Bu tür sistemlerde zamanlama (scheduling), bağlam değiştirme (context switching), öncelik yönetimi ve bellek tahsisi deterministik olmalıdır. Aksi halde sistem, zaman kısıtlarını ihlal ederek işlevsel olarak hatalı kabul edilir.

FreeRTOS, açık kaynaklı, hafif ve taşınabilir bir RTOS çekirdeğidir. Özellikle gömülü sistemlerde yaygın olarak kullanılmakta olup; görev yönetimi, öncelik tabanlı sıralama, zamanlayıcılar ve senkronizasyon mekanizmaları sunar. FreeRTOS'un modüler yapısı, kullanıcıların çekirdek üzerinde özelleştirilmiş zamanlama yaklaşımları geliştirmesine olanak tanır.

Bu projede FreeRTOS çekirdeği, POSIX tabanlı bir PC ortamında çalıştırılarak gerçek zamanlı görev zamanlama davranışları simüle edilmiştir. Amaç, RTOS'larda kullanılan çok seviyeli öncelikli zamanlama şemalarının teorik temellerini pratik bir uygulama üzerinden incelemektir.

2. Amaç: Projenin Amacı ve Scheduler Tasarımı

Bu projenin temel amacı, FreeRTOS görev yönetim mekanizmalarını kullanarak dört seviyeli öncelikli bir görev sıralayıcı tasarlamak ve bu yapının davranışlarını gözlemlemektir. Tasarlanan scheduler aşağıdaki hedefleri gözetmektedir:

- Gerçek zamanlı görevlerin kesilmeden ve deterministik biçimde çalıştırılması,
- Kullanıcı görevleri için çok seviyeli geri beslemeli bir zamanlama politikasının uygulanması,
- Zaman kuantumu, öncelik düşürme ve askıya alma mekanizmalarının açık şekilde gözlemlenmesi.

Scheduler yapısı, FreeRTOS çekirdeğinin sunduğu görev soyutlaması üzerinde, ayrı bir "scheduler görevi" olarak tasarlanmıştır. Bu yaklaşım, zamanlama mantığını uygulama görevlerinden ayırarak sistemin modülerliğini ve anlaşılabilirliğini artırmaktadır.

3. Yöntem: FreeRTOS görev yönetimi ve kendi scheduler fonksiyonları

Bu projede izlenen yöntem, FreeRTOS'un sunduğu temel görev yönetimi altyapısı kullanılarak, çok düzeyli ve öncelik tabanlı bir görevlendirme (scheduler) mekanizmasının uygulama seviyesinde modellenmesine dayanmaktadır. Amaç, gerçek zamanlı sistemlerde kullanılan zamanlama politikalarının çalışma mantığını simüle etmek ve bu politikaların sistem davranışı üzerindeki etkilerini gözlemlemektir.

3.1 FreeRTOS Görev Yönetimi Altyapısı

FreeRTOS; görev oluşturma, askıya alma, devam ettirme ve silme işlemleri için deterministik bir çekirdek sunmaktadır. Bu projede FreeRTOS'un yerleşik zamanlayıcısı doğrudan kullanılmamış, görevlerin ne zaman ve hangi öncelikte çalıştırılacağına ilişkin kararlar uygulama seviyesinde geliştirilen özel scheduler tarafından verilmiştir. FreeRTOS yalnızca görevlerin yürütülmesi ve bağlam değiştirme mekanizmasını sağlayan bir altyapı olarak kullanılmıştır.

Her görev FreeRTOS içerisinde bağımsız bir task olarak tanımlanmış ve TaskHandle aracılığıyla kontrol edilmiştir. Görevlerin askıya alınması, devam ettirilmesi ve sonlandırılması FreeRTOS API fonksiyonları ile yönetilmiştir. Bu yapı, gerçek işletim sistemlerinde çekirdek zamanlayıcısı ile kullanıcı alanı zamanlama mantığının ayrımına benzer bir yaklaşım sunmaktadır.

3.2 Görev Modeli ve Durum Yönetimi

Sistemdeki görevler, FreeRTOS görev yapısına ek olarak uygulama seviyesinde tanımlanan bir görev bilgi yapısı ile temsil edilmiştir. Bu yapı; görev kimliği, varış zamanı, öncelik seviyesi, toplam ve kalan çalışma süresi ile görev durumunu içermektedir. Görev durumları; henüz gelmemiş, hazır, çalışıyor, askıda ve tamamlanmış olarak sınıflandırılmıştır.

Bu durum modeli sayesinde görevlerin yaşam döngüsü açık biçimde izlenebilmekte ve scheduler tarafından deterministik zamanlama kararları alınabilmektedir. Görev durumlarının net şekilde tanımlanması, gerçek zamanlı sistemlerde analiz ve hata ayıklama açısından kritik öneme sahiptir.

3.3 Çok Düzeyli Görevlendirme ve Öncelik Politikası

Uygulanan zamanlama yaklaşımı, çok düzeyli geri beslemeli kuyruk (MLFQ) algoritmasından esinlenmiştir. En yüksek öncelik seviyesi gerçek zamanlı görevlere ayrılmıştır. Bu görevler FCFS esasına göre çalıştırılmakta ve tamamlanana kadar kesilmemektedir.

Kullanıcı görevleri ise zaman paylaşımlı bir yapı ile yönetilmektedir. Görevler belirlenen zaman kuantumu boyunca çalıştırılmakta, kuantum süresi dolduğunda askıya alınarak daha düşük öncelikli kuyruğa taşınmaktadır. Bu mekanizma, işlemciyi uzun süre meşgul eden görevlerin önceliğinin düşürülmesini sağlayarak sistemde adaletli bir kaynak paylaşımı sunmaktadır. Aynı öncelik seviyesindeki görevlerde kuyruk sırası esas alınarak starvation riski azaltılmıştır.

3.4 Zaman Aşımı ve Kaynak Yönetimi

Uzun süre çalıştırılamayan görevler için zaman aşımı mekanizması uygulanmıştır. Belirli bir süre boyunca sistemde bekleyen ancak tamamlanamayan görevler sonlandırılarak sistemden çıkarılmıştır. Bu yaklaşım, gerçek zamanlı sistemlerdeki “deadline miss” kavramının sadeleştirilmiş bir modelidir.

Bellek ve diğer kaynaklar FreeRTOS tarafından görev oluşturma sırasında tahsis edilmekte, görev sonlandırıldığında ise geri kazanılmaktadır. Scheduler, görevlerin yaşam döngüsünü kontrol ederek gereksiz kaynak kullanımını önlemekte ve sistem kararlılığını korumaktadır.

3.5 Zamanın İlerletilmesi ve Deterministik Davranış

Sistem zamanı sabit uzunlukta zaman dilimleri ile ilerletilmekte ve her zaman adımında scheduler yeniden karar vermektedir. Bu yapı sayesinde sistem deterministik bir davranış sergilemekte, aynı giriş verileri ile her çalıştırmada aynı görev sıralaması elde edilmektedir. Deterministiklik, gerçek zamanlı işletim sistemlerinin temel gereksinimlerinden biridir.

Bu yöntemsel yaklaşım ile FreeRTOS’un sağladığı temel mekanizmalar kullanılarak, gerçek işletim sistemlerinde kullanılan çok düzeyli ve öncelik tabanlı zamanlama politikaları başarıyla modellenmiştir.

4. Uygulama: Kod Açıklamaları ve Çalışma Çıktısı

4.1 Sistem Yapısı ve Modüler Tasarım

Uygulama, FreeRTOS’un POSIX portu üzerinde çalışacak şekilde modüler bir yapıda tasarlanmıştır. Sistem; ana program, zamanlayıcı (scheduler) modülü ve çalışan görevler (worker tasks) olmak üzere üç temel bileşenden oluşmaktadır. Bu ayrım sayesinde zamanlama politikası, görevlerin gerçek yürütme mantığından ayrılmış ve sistemin anlaşılabilirliği artırılmıştır.

Ana program, FreeRTOS ortamını başlatmakta ve scheduler görevini sistemde en yüksek öncelikle çalıştırmaktadır. Scheduler modülü, görevlerin giriş dosyasından okunması, uygun kuyruklara yerleştirilmesi ve hangi görevin ne zaman çalıştırılacağına karar verilmesinden sorumludur. Worker görevleri ise yalnızca kendilerine verilen zaman dilimlerinde çalışarak scheduler’ın verdiği kararları uygular.

4.2 Görevlerin Oluşturulması ve Yönetimi

Görevler, dışarıdan sağlanan metin dosyasından okunarak sisteme tanıtılmaktadır. Her görev için varış zamanı, öncelik seviyesi ve çalışma süresi bilgileri alınmakta; görev zamanı geldiğinde FreeRTOS üzerinde ilgili worker görevi oluşturulmaktadır. Ancak görev oluşturulduktan sonra doğrudan çalıştırılmamakta, scheduler tarafından uygun görüldüğü ana kadar askıda tutulmaktadır.

Görevlerin askıya alınması, devam ettirilmesi ve sonlandırılması işlemleri FreeRTOS API fonksiyonları kullanılarak gerçekleştirilmektedir. Böylece scheduler, görevlerin yürütülmesini doğrudan kontrol edebilmekte ve çok düzeyli zamanlama politikasını yazılım seviyesinde uygulayabilmektedir.

4.3 Zamanlama Kararları ve Çalışma Mantığı

Gerçek zamanlı görevler (öncelik 0), sisteme girdikleri anda FCFS esasına göre çalıştırılmakta ve tamamlanana kadar kesilmemektedir. Bu sırada daha düşük öncelikli görevler askıya alınmaktadır. Kullanıcı görevleri ise çok düzeyli geri beslemeli kuyruk yapısına göre zamanlanmaktadır.

Kullanıcı görevleri belirlenen zaman kuantumu boyunca çalıştırılmakta, kuantum süresi dolduğunda görev askıya alınarak önceliği düşürülmekte ve uygun kuyruğa yeniden yerleştirilmektedir. Aynı öncelik seviyesindeki görevlerde kuyruk sırası dikkate alınarak adil bir yürütme sağlanmaktadır. Uzun süre çalıştırılamayan görevler için zaman aşımı kontrolü uygulanarak sistemden çıkarılmaktadır.

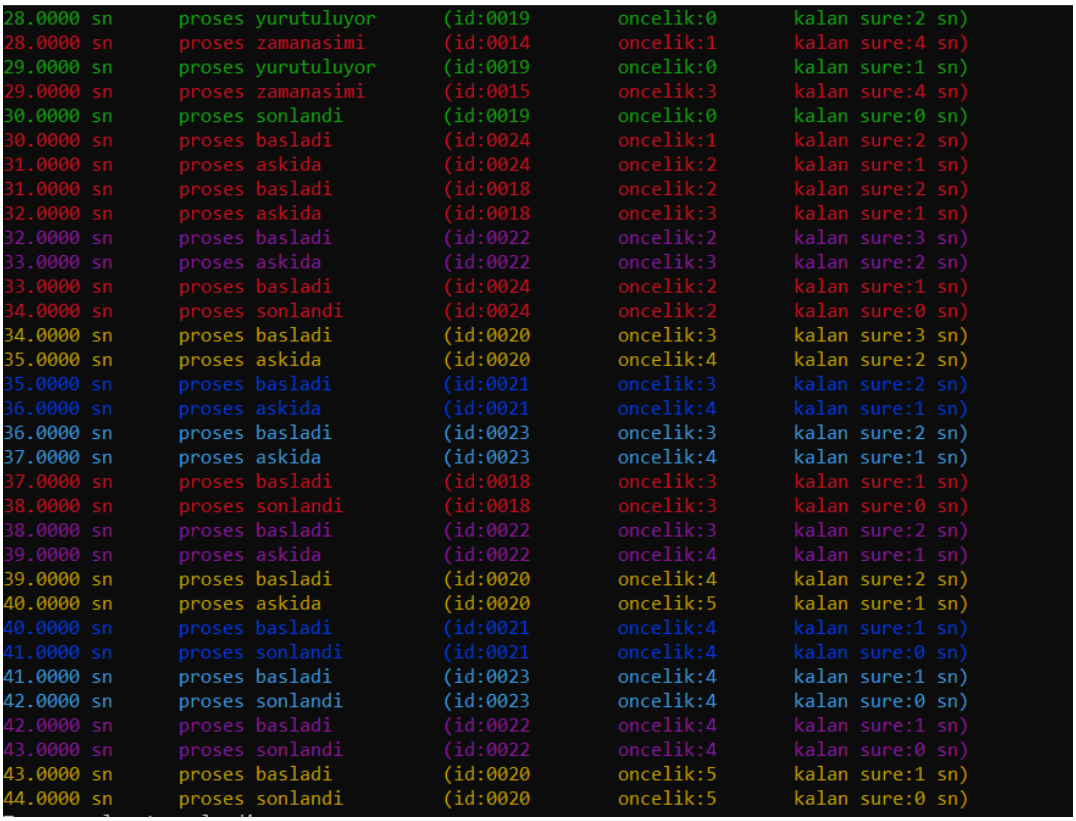
4.4 Terminal Çıktıları ve Gözlemlenen Davranış

Sistemin çalışması sırasında scheduler tarafından alınan tüm kararlar terminal çıktıları aracılığıyla gözlemlenebilmektedir. Görevlerin başlatılması, askıya alınması, devam ettirilmesi ve sonlandırılması anlarında bilgilendirici mesajlar üretilmektedir. Her görev için farklı renklerin kullanılması, görevlerin zaman içerisindeki yürütme dilimlerinin kolayca ayırt edilmesini sağlamaktadır.

Terminal çıktıları incelendiğinde, gerçek zamanlı görevlerin her zaman öncelikli olarak çalıştırıldığı, kullanıcı görevlerinin ise zaman paylaşımlı ve öncelik düşürmeli bir yapıda yürütüldüğü açıkça görülmektedir. Bu çıktılar, geliştirilen scheduler'ın tasarlanan zamanlama politikasını doğru biçimde uyguladığını göstermektedir.

Aşağıda program çıktısı yer almaktadır.

```
ramazan@DESKTOP-8OAQ7J3:/mnt/c/Users/ramaz/OneDrive/Desktop/isletim_sistemleri_projesi$ ./freertos_sim giris.txt
Okunan gorev sayisi: 25
0.0000 sn       proses basladi          (id:0000        oncelik:1       kalan sure:2 sn)
1.0000 sn       proses askida           (id:0000        oncelik:2       kalan sure:1 sn)
1.0000 sn       proses basladi          (id:0001        oncelik:0       kalan sure:1 sn)
2.0000 sn       proses sonlandi         (id:0001        oncelik:0       kalan sure:0 sn)
2.0000 sn       proses basladi          (id:0003        oncelik:0       kalan sure:3 sn)
3.0000 sn       proses yurutuluyor      (id:0003        oncelik:0       kalan sure:2 sn)
4.0000 sn       proses yurutuluyor      (id:0003        oncelik:0       kalan sure:1 sn)
5.0000 sn       proses sonlandi         (id:0003        oncelik:0       kalan sure:0 sn)
5.0000 sn       proses basladi          (id:0006        oncelik:0       kalan sure:4 sn)
6.0000 sn       proses yurutuluyor      (id:0006        oncelik:0       kalan sure:3 sn)
7.0000 sn       proses yurutuluyor      (id:0006        oncelik:0       kalan sure:2 sn)
8.0000 sn       proses yurutuluyor      (id:0006        oncelik:0       kalan sure:1 sn)
9.0000 sn       proses sonlandi         (id:0006        oncelik:0       kalan sure:0 sn)
9.0000 sn       proses basladi          (id:0007        oncelik:0       kalan sure:4 sn)
10.0000 sn      proses yurutuluyor      (id:0007        oncelik:0       kalan sure:3 sn)
11.0000 sn      proses yurutuluyor      (id:0007        oncelik:0       kalan sure:2 sn)
12.0000 sn      proses yurutuluyor      (id:0007        oncelik:0       kalan sure:1 sn)
13.0000 sn      proses sonlandi         (id:0007        oncelik:0       kalan sure:0 sn)
13.0000 sn      proses basladi          (id:0008        oncelik:0       kalan sure:2 sn)
14.0000 sn      proses yurutuluyor      (id:0008        oncelik:0       kalan sure:1 sn)
15.0000 sn      proses sonlandi         (id:0008        oncelik:0       kalan sure:0 sn)
15.0000 sn      proses basladi          (id:0010        oncelik:0       kalan sure:3 sn)
16.0000 sn      proses yurutuluyor      (id:0010        oncelik:0       kalan sure:2 sn)
17.0000 sn      proses yurutuluyor      (id:0010        oncelik:0       kalan sure:1 sn)
18.0000 sn      proses sonlandi         (id:0010        oncelik:0       kalan sure:0 sn)
18.0000 sn      proses basladi          (id:0016        oncelik:0       kalan sure:4 sn)
19.0000 sn      proses yurutuluyor      (id:0016        oncelik:0       kalan sure:3 sn)
20.0000 sn      proses yurutuluyor      (id:0016        oncelik:0       kalan sure:2 sn)
21.0000 sn      proses yurutuluyor      (id:0016        oncelik:0       kalan sure:1 sn)
21.0000 sn      proses zamanasimi       (id:0000        oncelik:2       kalan sure:1 sn)
21.0000 sn      proses zamanasimi       (id:0002        oncelik:3       kalan sure:2 sn)
21.0000 sn      proses zamanasimi       (id:0004        oncelik:2       kalan sure:2 sn)
22.0000 sn      proses sonlandi         (id:0016        oncelik:0       kalan sure:0 sn)
22.0000 sn      proses basladi          (id:0017        oncelik:0       kalan sure:4 sn)
22.0000 sn      proses zamanasimi       (id:0005        oncelik:2       kalan sure:3 sn)
23.0000 sn      proses yurutuluyor      (id:0017        oncelik:0       kalan sure:3 sn)
24.0000 sn      proses yurutuluyor      (id:0017        oncelik:0       kalan sure:2 sn)
24.0000 sn      proses zamanasimi       (id:0009        oncelik:2       kalan sure:4 sn)
25.0000 sn      proses yurutuluyor      (id:0017        oncelik:0       kalan sure:1 sn)
25.0000 sn      proses zamanasimi       (id:0011        oncelik:3       kalan sure:2 sn)
26.0000 sn      proses sonlandi         (id:0017        oncelik:0       kalan sure:0 sn)
26.0000 sn      proses basladi          (id:0019        oncelik:0       kalan sure:4 sn)
26.0000 sn      proses zamanasimi       (id:0012        oncelik:3       kalan sure:2 sn)
26.0000 sn      proses zamanasimi       (id:0013        oncelik:1       kalan sure:2 sn)
27.0000 sn      proses yurutuluyor      (id:0019        oncelik:0       kalan sure:3 sn)
28.0000 sn      proses yurutuluyor      (id:0019        oncelik:0       kalan sure:2 sn)
28.0000 sn      proses zamanasimi       (id:0014        oncelik:1       kalan sure:4 sn)
29.0000 sn      proses yurutuluyor      (id:0019        oncelik:0       kalan sure:1 sn)
```



5. Sonuç ve Bilimsel Değerlendirme

Bu çalışmada geliştirilen çok düzeyli ve öncelik tabanlı görev sıralayıcısı, gerçek zamanlı işletim sistemlerinde kullanılan zamanlama şemalarının temel ilkelerini kavramsal ve deneysel olarak inceleme olanağı sunmuştur. FreeRTOS'un deterministik görev yönetim altyapısı, uygulama seviyesinde tasarlanan scheduler ile birlikte kullanılarak, farklı öncelik sınıflarına sahip görevlerin sistem davranışı üzerindeki etkileri gözlemlenmiştir.

5.1 Çok Düzeyli Görevlendirme Şemalarının Gerekçesi

Gerçek işletim sistemlerinde tek seviyeli zamanlama algoritmaları, karmaşık ve heterojen iş yüklerini yönetmekte yetersiz kalmaktadır. Özellikle hem zaman kritik (real-time) hem de en iyi çaba (best-effort) görevlerin birlikte çalıştığı sistemlerde, farklı zamanlama politikalarının bir arada kullanılması zorunlu hale gelmektedir. Bu nedenle modern işletim sistemleri, çok düzeyli ve öncelik tabanlı görevlendirme şemalarını tercih etmektedir.

Bu projede kullanılan çok düzeyli geri beslemeli yaklaşım, Linux gibi genel amaçlı işletim sistemlerinde kullanılan zamanlayıcıların sadeleştirilmiş bir modelidir. Gerçek zamanlı görevler için deterministik ve kesintisiz yürütme sağlanırken, kullanıcı görevleri için zaman paylaşımlı ve öncelik düşürmeli bir yapı uygulanmıştır. Bu sayede hem zaman garantisi gerektiren görevler hem de sistem adaletini gözeten görevler aynı platform üzerinde yönetilebilmiştir.

5.2 Gerçek İşletim Sistemleri ile Karşılaştırmalı Analiz

Gerçek İşletim Sistemleri ile Karşılaştırma ve Eksiklikler

Linux çekirdeğindeki Completely Fair Scheduler (CFS) veya RTOS'lardaki Fixed Priority Pre-emptive Scheduling ile kıyaslandığında, tasarımımız eğitimsel amaçlı bazı sadeleştirmeler içermektedir.

Mevcut Eksiklikler ve İyileştirme Önerileri:

1. **Arama Karmaşıklığı:** Mevcut tasarımda selectNextTask fonksiyonu, uygun görevi bulmak için tüm görev listesini taramaktadır (O(N) karmaşıklığı). Gerçek sistemlerde bu işlem, *Linked List* veya *Bitmap* yapıları kullanılarak O(1) süresinde yapılır. İyileştirme olarak görevler, önceliklerine göre ayrılmış bağlı listelerde tutulabilir.

2. **Bağlam Değiştirme (Context Switch) Maliyeti:** Her saniye yapılan zorunlu görev değişimi, işlemci üzerinde ek yük oluşturmaktadır. Dinamik kuantum süreleri (önceliğe göre değişen süreler) kullanılarak bu maliyet düşürülebilir.

3. **Çok Çekirdek Desteği:** Tasarım tek işlemcili bir sistemi simüle etmektedir, yük dengeleme (load balancing) mekanizmaları eklenmemiştir.

5.3 Bellek ve Kaynak Ayırma Şemalarının Değerlendirilmesi

Gerçek zamanlı sistemlerde bellek ve kaynak ayırma politikaları, zamanlama kadar kritik öneme sahiptir. Dinamik bellek tahsisi, belirsiz gecikmelere yol açabileceğinden birçok RTOS'ta sınırlı veya kontrollü biçimde kullanılmaktadır. Bu projede görevlerin yaşam döngüsü FreeRTOS tarafından yönetilmiş, görev sonlandırıldığında bellek ve diğer kaynaklar otomatik olarak geri kazanılmıştır.

Scheduler tarafından kullanılan görev bilgi yapıları, gerçek işletim sistemlerindeki Görev Kontrol Bloğu (TCB) kavramına benzer şekilde çalışmaktadır. Bu yapılar sayesinde görevlerin durumu, önceliği ve kalan çalışma süresi izlenebilmekte; kaynakların yalnızca aktif görevler için tahsis edilmesi sağlanmaktadır. Bu yaklaşım, sistemin bellek kullanımında öngörülebilirlik ve kararlılık sunmaktadır.

5.4 Sınırlamalar ve Olası İyileştirmeler

Geliştirilen çok düzeyli görevlendirme şeması, zamanlama politikalarının temel prensiplerini ortaya koymakla birlikte bazı sınırlamalara sahiptir. Öncelik seviyelerinin sabit olması, görevler için açık bir son teslim zamanı (deadline) modelinin bulunmaması ve bellek kullanımının ayrıntılı olarak izlenmemesi bu sınırlamalar arasında yer almaktadır.

İlerleyen çalışmalarda, en erken son teslim zamanına göre zamanlama (EDF) gibi algoritmaların eklenmesi, dinamik öncelik ayarlama mekanizmalarının uygulanması ve bellek tahsisinin daha sıkı kontrol altına alınması sistemin gerçek işletim sistemlerine daha fazla yaklaşmasını sağlayacaktır. Ayrıca çok çekirdekli mimariler için görev dağıtımı ve yük dengeleme algoritmalarının eklenmesi, çalışmanın kapsamını genişletebilir.

5.5 Genel Bilimsel Değerlendirme

Sonuç olarak bu çalışma, FreeRTOS kullanılarak çok düzeyli ve öncelik tabanlı görev sıralama şemalarının nasıl uygulanabileceğini ve bu şemaların sistem davranışı üzerindeki etkilerini ortaya koymuştur. Elde edilen bulgular, çok düzeyli görevlendirmenin neden modern işletim sistemlerinde yaygın olarak kullanıldığını açıklamakta ve FreeRTOS'un hem gömülü sistemler hem de akademik amaçlı simülasyonlar için etkin bir platform olduğunu göstermektedir.

6.KAYNAKÇA

1-FreeRTOS. (2024). *The FreeRTOS Reference Manual*.

2-Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating System Concepts* (10th ed.). Wiley.