

İŞLETİM SİSTEMLERİ

*FreeRTOS Üzerinde Çok Düzeyli Geri Beslemeli Kuyruk (MLFQ)
Zamanlayıcı Tasarımı ve Simülasyonu*

EKİP ÜYELERİ : Alpcan Ekşi - B231210020
Ümit Çetin - B231210070
Koray Temizkan - B231210010
Hüseyin Yiğci - B231210088
Yusuf Ziya Gök - B231210046

GitHub Bağlantısı : <https://github.com/cetinumit/akhuz>

Giriş: FreeRTOS ve Gerçek Zamanlı Sistemler

Gömülü sistemlerin karmaşıklığının artmasıyla birlikte, "Super-Loop" (sonsuz döngü) mimarileri, zaman kritik görevlerin deterministik yönetiminde yetersiz kalmaktadır. Bu projede, açı kaynaklı ve endüstri standarı bir Gerçek Zamanlı İşletim Sistemi (RTOS) olan **FreeRTOS** kullanılarak çoklu görev yönetimi (multitasking) hedeflenmiştir.

Gerçek zamanlı işletim sistemleri, işlemlerin sadece mantıksal doğruluğunu değil, aynı zamanda belirli bir zaman dilimi içinde tamamlanmasını da garanti eder. Bu raporda, mikrodenetleyicinin (CPU, Bellek) verimli kullanımı, görevler arası haberleşme ve öncelik tabanlı zamanlama (Priority-Based Scheduling) mekanizmaları incelenmiştir.

Amaç ve Yöntem

Amaç

Projenin temel amacı, kısıtlı kaynaklara sahip bir mikrodenetleyici üzerinde, öncelik tabanlı ve önleyici (preemptive) bir zamanlayıcı yapısı kurmaktır. Özel amaçlar şunlardır:

- Farklı önceliklere sahip görevlerin, deterministik bir yapıda eşzamanlı (concurrency) çalıştırılması.
- Görevler arası veri bütünlüğünü koruyarak haberleşmenin sağlanması (Inter-task Communication).
- FreeRTOS çekirdeğinin (Kernel) kaynak yönetimi yeteneklerinin deneyimsel olarak doğrulanması.

Yöntem ve Teknik Altyapı

Bu bölümde, FreeRTOS'un görev yönetimi ve kaynak tahsis'i için kullandığı veri yapıları ve yöntemler teknik detaylarıyla açıklanmıştır.

3. Kaynak Yönetimi ve Kullanılan Yapılar

Görevlendirici (Scheduler) tarafından bellek ve kaynakların yönetimi için aşağıdaki kritik yapılar kullanılmıştır:

- TCB (Task Control Block):** Her görev oluşturulduğunda, FreeRTOS dinamik veya statik olarak bir TCB tahsis eder. TCB; görevin yığın (stack) başlangıç adresini, öncelik seviyesini, durumunu (Ready, Blocked, Suspended) ve yığın üzerindeki son konumunu (Stack Pointer) tutan, işletim sisteminin o görev'e dair "kimlik kartı"dır.
- Hazır Listeleri (Ready Lists):** Scheduler, `pxReadyTasksLists` adı verilen bir dizi liste kullanır. Her öncelik seviyesi için ayrı bir liste bulunur. Zamanlayıcı, her zaman en yüksek önceliğe sahip listedeki ilk görevi CPU'ya tahsis eder.
- Kuyruklar (Queues):** Görevler arasında veri transferi için thread-safe (iş parçacığı güvenli) bir yapı olan kuyruklar kullanılmıştır. Veriler kuyruğa **değer ile kopyalanarak (copy by value)** gönderilir. Bu, kaynak görevin veriyi gönderdikten sonra yerel değişkenini güvenle değiştirebilmesini sağlar.
- Heap Yönetimi (Bellek Tahsisi):** Görevler ve kuyruklar oluşturulurken `pvPortMalloc()` fonksiyonu ile Heap alanında bellek ayrıılır. Projede, bellek fragmentasyonunu minimize etmek için `Heap_4.c` (veya kullanılan hangisiyse) şeması tercih edilmiştir. Bu şema, bitişik boş blokları birleştirerek verimli bir bellek yönetimi sunar.

Uygulama ve Program Yapısı

Program, modüler bir yapıda tasarlanmış olup ana işlevler ve arayüzler aşağıda açıklanmıştır:

Genel Yapı ve Modüller :

Yazılım mimarisi üç ana katmandan oluşmaktadır:

- 1. Donanım Soyutlama Katmanı (HAL):** Mikrodenetleyici çevre birimlerinin (GPIO, UART, vb.) başlatıldığı alt seviye sürücüler.
- 2. Kernel (Çekirdek) Katmanı:** FreeRTOS API'leri ile görevlerin oluşturulduğu ve zamanlayıcının başlatıldığı katman.
- 3. Uygulama Katmanı (Tasks):** Sistemin asıl işini yapan görev fonksiyonları.

Ana İşlevler ve Arayüzler :

Projede kullanılan temel fonksiyonların işlevleri ve gerekçeleri şöyledir:

- **main() Fonksiyonu:** Sistemin giriş noktasıdır. Önce [SystemInit] ile saat ayarlarını yapar, ardından donanım birimlerini başlatır. En kritik görevi; xTaskCreate ile görevleri TCB'lere bağlamak ve vTaskStartScheduler ile RTOS çekirdeğini başlatmaktadır. Bu noktadan sonra kontrol tamamen işletim sistemine geçer.
- **Görev Fonksiyonları ([Örn: SensorTask], [Örn: DisplayTask]):** Bu fonksiyonlar sonsuz bir döngü (for(;;)) içerisinde çalışır.
 - *Gerekçe:* Bir RTOS görevi asla return ile dönemez; ya sonsuz döngüde kalmalı ya da vTaskDelete ile kendini yok etmelidir.
- **Veri İletim Arayüzü (xQueueSend, xQueueReceive):** Görevler arası senkronizasyon için kullanılır.
 - *Gerekçe:* Global değişken kullanımı yerine kuyruk yapısının seçilmesi, "Race Condition" (Yarış Durumu) riskini ortadan kaldırmak ve veri bütünlüğünü sağlamak içindir.

```
// --- İŞÇİ TASK (WORKER) ---
// Bu task sadece Scheduler izin verdiğiinde 1 sn (simüle) çalışır.
void GeneralTaskFunction(void* pvParameters) {
    TaskHandle* t = (TaskHandle*) pvParameters;

    while (1) {
        // 1) 1 saniyelik iş: kalan süreyi azalt
        if (t->remainingTime > 0) {
            t->remainingTime--;
        }

        // 2) Controller'a "1 saniye bitti" diye haber ver
        xTaskNotifyGive(xSchedulerTaskHandle);

        // 3) Controller tekrar çağrırama kadar kendimi askıya al
        vTaskSuspend(NULL);
    }
}
```

```
void init_scheduler() {
    // Kuyrukları task pointeri tutacak şekilde oluştur
    xQueueRealTime = xQueueCreate(50, sizeof(TaskHandle*));
    xQueueHigh     = xQueueCreate(50, sizeof(TaskHandle*));
    xQueueMedium   = xQueueCreate(50, sizeof(TaskHandle*));
    xQueueLow      = xQueueCreate(50, sizeof(TaskHandle*));

    // Controller en yüksek önceliğe sahip olmalı
    xTaskCreate(SchedulerController,
                "Controller",
                configMINIMAL_STACK_SIZE * 4,
                NULL,
                configMAX_PRIORITIES - 1,
                NULL);
}
```

Terminal Çıktıları

```
koray@KoraysLinuxPC:~/Desktop/akhuz-main (copy)$ ./freertos_sim giris.txt
--- Dosya Okuma Basladi: giris.txt ---
Okundu -> Task: T0 | Gelis: 0 | Oncelik: 1 | Sure: 2
Okundu -> Task: T1 | Gelis: 1 | Oncelik: 0 | Sure: 1
Okundu -> Task: T2 | Gelis: 1 | Oncelik: 3 | Sure: 2
Okundu -> Task: T3 | Gelis: 1 | Oncelik: 0 | Sure: 3
Okundu -> Task: T4 | Gelis: 1 | Oncelik: 2 | Sure: 2
Okundu -> Task: T5 | Gelis: 2 | Oncelik: 2 | Sure: 3
Okundu -> Task: T6 | Gelis: 2 | Oncelik: 0 | Sure: 4
Okundu -> Task: T7 | Gelis: 2 | Oncelik: 0 | Sure: 4
Okundu -> Task: T8 | Gelis: 3 | Oncelik: 0 | Sure: 2
Okundu -> Task: T9 | Gelis: 4 | Oncelik: 2 | Sure: 4
Okundu -> Task: T10 | Gelis: 5 | Oncelik: 0 | Sure: 3
Okundu -> Task: T11 | Gelis: 5 | Oncelik: 3 | Sure: 2
Okundu -> Task: T12 | Gelis: 6 | Oncelik: 3 | Sure: 2
Okundu -> Task: T13 | Gelis: 6 | Oncelik: 1 | Sure: 2
Okundu -> Task: T14 | Gelis: 8 | Oncelik: 1 | Sure: 4
Okundu -> Task: T15 | Gelis: 9 | Oncelik: 3 | Sure: 4
Okundu -> Task: T16 | Gelis: 11 | Oncelik: 0 | Sure: 4
Okundu -> Task: T17 | Gelis: 12 | Oncelik: 0 | Sure: 4
Okundu -> Task: T18 | Gelis: 14 | Oncelik: 2 | Sure: 2
Okundu -> Task: T19 | Gelis: 15 | Oncelik: 0 | Sure: 4
Okundu -> Task: T20 | Gelis: 16 | Oncelik: 3 | Sure: 3
Okundu -> Task: T21 | Gelis: 18 | Oncelik: 3 | Sure: 2
Okundu -> Task: T22 | Gelis: 22 | Oncelik: 2 | Sure: 3
Okundu -> Task: T23 | Gelis: 23 | Oncelik: 3 | Sure: 2
Okundu -> Task: T24 | Gelis: 24 | Oncelik: 1 | Sure: 2
--- Toplam 25 task yüklandı. ---

[SISTEM] FreeRTOS Kernel Baslatiliyor... (Simulasyon Basladi)
[DIAGNOSTIC] vTaskStartScheduler çağrılacak...
Simulasyon Baslıyor...
0.0000 sn proses başladi (id:0000 onceli:1 kalan sure:2 sn)
1.0000 sn proses askıda (id:0000 onceli:2 kalan sure:1 sn)
1.0000 sn proses başladi (id:0001 onceli:0 kalan sure:1 sn)
2.0000 sn proses sonlandı (id:0001 onceli:0 kalan sure:0 sn)
2.0000 sn proses başladi (id:0003 onceli:0 kalan sure:3 sn)
3.0000 sn proses yürütülüyor (id:0003 onceli:0 kalan sure:2 sn)
4.0000 sn proses yürütülüyor (id:0003 onceli:0 kalan sure:1 sn)
5.0000 sn proses sonlandı (id:0003 onceli:0 kalan sure:0 sn)
5.0000 sn proses başladi (id:0006 onceli:0 kalan sure:4 sn)
6.0000 sn proses yürütülüyor (id:0006 onceli:0 kalan sure:3 sn)
7.0000 sn proses yürütülüyor (id:0006 onceli:0 kalan sure:2 sn)
8.0000 sn proses yürütülüyor (id:0006 onceli:0 kalan sure:1 sn)
9.0000 sn proses sonlandı (id:0006 onceli:0 kalan sure:0 sn)
9.0000 sn proses başladi (id:0007 onceli:0 kalan sure:4 sn)
10.0000 sn proses yürütülüyor (id:0007 onceli:0 kalan sure:3 sn)
11.0000 sn proses yürütülüyor (id:0007 onceli:0 kalan sure:2 sn)
12.0000 sn proses yürütülüyor (id:0007 onceli:0 kalan sure:1 sn)
13.0000 sn proses sonlandı (id:0007 onceli:0 kalan sure:0 sn)
13.0000 sn proses başladi (id:0008 onceli:0 kalan sure:2 sn)
14.0000 sn proses yürütülüyor (id:0008 onceli:0 kalan sure:1 sn)
15.0000 sn proses sonlandı (id:0008 onceli:0 kalan sure:0 sn)
15.0000 sn proses başladi (id:0010 onceli:0 kalan sure:3 sn)
16.0000 sn proses yürütülüyor (id:0010 onceli:0 kalan sure:2 sn)
17.0000 sn proses yürütülüyor (id:0010 onceli:0 kalan sure:1 sn)
18.0000 sn proses sonlandı (id:0010 onceli:0 kalan sure:0 sn)
18.0000 sn proses başladi (id:0016 onceli:0 kalan sure:4 sn)
19.0000 sn proses yürütülüyor (id:0016 onceli:0 kalan sure:3 sn)

18.0000 sn proses başladi (id:0016 onceli:0 kalan sure:4 sn)
19.0000 sn proses yürütülüyor (id:0016 onceli:0 kalan sure:3 sn)
20.0000 sn proses yürütülüyor (id:0016 onceli:0 kalan sure:2 sn)
21.0000 sn proses zamanasımı (id:0000 onceli:2 kalan sure:1 sn)
21.0000 sn proses zamanasımı (id:0002 onceli:3 kalan sure:2 sn)
21.0000 sn proses zamanasımı (id:0004 onceli:2 kalan sure:2 sn)
21.0000 sn proses yürütülüyor (id:0016 onceli:0 kalan sure:1 sn)
22.0000 sn proses sonlandı (id:0016 onceli:0 kalan sure:0 sn)
22.0000 sn proses zamanasımı (id:0005 onceli:2 kalan sure:3 sn)
22.0000 sn proses başladi (id:0017 onceli:0 kalan sure:4 sn)
23.0000 sn proses yürütülüyor (id:0017 onceli:0 kalan sure:3 sn)
24.0000 sn proses zamanasımı (id:0009 onceli:2 kalan sure:4 sn)
24.0000 sn proses yürütülüyor (id:0017 onceli:0 kalan sure:2 sn)
25.0000 sn proses zamanasımı (id:0011 onceli:3 kalan sure:2 sn)
25.0000 sn proses yürütülüyor (id:0017 onceli:0 kalan sure:1 sn)
26.0000 sn proses sonlandı (id:0017 onceli:0 kalan sure:0 sn)
26.0000 sn proses zamanasımı (id:0012 onceli:3 kalan sure:2 sn)
26.0000 sn proses zamanasımı (id:0013 onceli:1 kalan sure:2 sn)
26.0000 sn proses başladi (id:0019 onceli:0 kalan sure:4 sn)
27.0000 sn proses yürütülüyor (id:0019 onceli:0 kalan sure:3 sn)
28.0000 sn proses zamanasımı (id:0014 onceli:1 kalan sure:4 sn)
28.0000 sn proses yürütülüyor (id:0019 onceli:0 kalan sure:2 sn)
29.0000 sn proses zamanasımı (id:0015 onceli:3 kalan sure:4 sn)
29.0000 sn proses yürütülüyor (id:0019 onceli:0 kalan sure:1 sn)
30.0000 sn proses sonlandı (id:0019 onceli:0 kalan sure:0 sn)
30.0000 sn proses başladi (id:0024 onceli:1 kalan sure:2 sn)
31.0000 sn proses askıda (id:0024 onceli:2 kalan sure:1 sn)
31.0000 sn proses başladi (id:0018 onceli:2 kalan sure:2 sn)
32.0000 sn proses askıda (id:0018 onceli:3 kalan sure:1 sn)
32.0000 sn proses başladi (id:0022 onceli:2 kalan sure:3 sn)
33.0000 sn proses askıda (id:0022 onceli:3 kalan sure:2 sn)
33.0000 sn proses yürütülüyor (id:0024 onceli:2 kalan sure:1 sn)
34.0000 sn proses sonlandı (id:0024 onceli:2 kalan sure:0 sn)
34.0000 sn proses başladi (id:0020 onceli:3 kalan sure:3 sn)
35.0000 sn proses askıda (id:0020 onceli:4 kalan sure:2 sn)
35.0000 sn proses başladi (id:0021 onceli:3 kalan sure:2 sn)
36.0000 sn proses askıda (id:0021 onceli:4 kalan sure:1 sn)
36.0000 sn proses başladi (id:0023 onceli:3 kalan sure:2 sn)
37.0000 sn proses askıda (id:0023 onceli:4 kalan sure:1 sn)
37.0000 sn proses yürütülüyor (id:0018 onceli:3 kalan sure:1 sn)
38.0000 sn proses sonlandı (id:0018 onceli:3 kalan sure:0 sn)
38.0000 sn proses yürütülüyor (id:0022 onceli:3 kalan sure:2 sn)
39.0000 sn proses askıda (id:0022 onceli:4 kalan sure:1 sn)
39.0000 sn proses yürütülüyor (id:0020 onceli:4 kalan sure:2 sn)
40.0000 sn proses askıda (id:0020 onceli:5 kalan sure:1 sn)
40.0000 sn proses yürütülüyor (id:0021 onceli:4 kalan sure:1 sn)
41.0000 sn proses sonlandı (id:0021 onceli:4 kalan sure:0 sn)
41.0000 sn proses yürütülüyor (id:0023 onceli:4 kalan sure:1 sn)
42.0000 sn proses sonlandı (id:0023 onceli:4 kalan sure:0 sn)
42.0000 sn proses yürütülüyor (id:0022 onceli:4 kalan sure:1 sn)
43.0000 sn proses sonlandı (id:0022 onceli:4 kalan sure:0 sn)
43.0000 sn proses yürütülüyor (id:0020 onceli:5 kalan sure:1 sn)
44.0000 sn proses sonlandı (id:0020 onceli:5 kalan sure:0 sn)

Simulasyon Bitti.
```

NOT : Proje Linux ve Macos ortamında aşağıdaki adımlar izlenerek çalıştırılabilir olmaktadır:

1. Proje klasörüne girilir.
2. Derleme işlemi make komutu ile gerçekleştirilir.
3. Program ./freertos_sim giris.txt komutu ile çalıştırılır.

Windows işletim sisteminde ise gerekli Linux kütüphanelerinin (POSIX, pthread vb.) desteklenmesi amacıyla Windows Subsystem for Linux (WSL) kullanılması gerekmektedir. Proje, WSL üzerinde Ubuntu dağıtımını kullanılarak sorunsuz şekilde derlenmiş ve çalıştırılmıştır.

Tartışma ve Karşılaştırma

Bu bölümde, uygulanan RTOS yapısı, GPOS (General Purpose OS - Linux/Windows gibi) sistemlerle karşılaştırılmış ve analizi yapılmıştır.

Çok Düzeyli Görevlendirme (Multilevel Scheduling) ve Gerekçesi

Uygulamamızda kullanılan "Preemptive Priority-Based Scheduling" (Öncelik Tabanlı Kesici Zamanlama), sistemin deterministik olmasını sağlar. Ancak "Gerçek" (GPOS) işletim sistemleri genellikle CFS (Completely Fair Scheduler) veya dinamik öncelikli çok seviyeli geri besleme kuyrukları (Multilevel Feedback Queue) kullanır.

- **Neden RTOS Şeması Kullanıldı?** GPOS sistemler "iş hacmini" (throughput) maksimize etmeye odaklanırken, bizim projemiz "gecikmeyi" (latency) minimize etmeye odaklanır. Çok düzeyli bir şemada, yüksek öncelikli bir yanık alarmı sensörü verisi, düşük öncelikli bir ekran güncelleme işlemini arasında kesmelidir. Bu deterministik davranış, ancak RTOS'un katı bir şemasıyla mümkündür.

Eksiklikler ve İyileştirme Önerileri

Mevcut tasarımda karşılaşılabilecek potansiyel sorunlar ve çözümler şunlardır:

1. **Öncelik Dönmesi (Priority Inversion):** Düşük öncelikli bir görev, yüksek öncelikli bir görevin ihtiyaç duyduğu bir kaynağı (örn. Semaphor) tutarsa, sistem kilitlenebilir.
 - *İyileştirme:* "Priority Inheritance" (Öncelik Mirası) mekanizması aktif edilmelidir. Bu sayede kaynağı tutan düşük öncelikli görevin önceliği geçici olarak yükseltilir.
2. **Açlık (Starvation):** Sürekli çalışan yüksek öncelikli görevler, düşük öncelikli görevlerin hiç CPU zamanı alamamasına neden olabilir.
 - *İyileştirme:* GPOS'larda kullanılan "Aging" (Yaşlandırma) tekniği uygulanabilir veya FreeRTOS'un `vTaskDelay` fonksiyonu ile yüksek öncelikli görevlere zorunlu beklenme süreleri eklenerek CPU bırakılabilir.
3. **Bellek Yönetimi: `Hheap_4`** dinamik tahsis kullansa da, uzun çalışma sürelerinde bellek parçalanması (fragmentation) riski taşır.
 - *İyileştirme:* Kritik sistemlerde "Static Allocation" (Statik Tahsis) kullanılarak TCB ve Yığın bellekleri derleme zamanında ayrılmalıdır.

SONUÇ

Bu projede, FreeRTOS kullanılarak gömülü bir sistem üzerinde çoklu görev yapısı başarıyla kurgulanmıştır. Deneysel sonuçlar, öncelik tabanlı zamanlayıcının, görevleri tanımlanan önceliklere uygun olarak yönettiğini göstermiştir. Kuyruk mekanizması sayesinde görevler arası veri传递, veri kaybı olmadan sağlanmıştır.

Sonuç olarak; FreeRTOS'un sağladığı soyutlama katmanı, karmaşık zamanlama algoritmalarını geliştirici için şeffaf hale getirmiştir, ancak geliştiricinin "Deadlock" ve "Starvation" gibi işlevleri kavramlarına hakim olmasını zorunlu kılmıştır. Sistem, deterministik davranış gereksinimlerini karşılamaktadır.