**logo, ticari marka, amblem, simge, sembol içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**BLM0230 BİLGİSAYAR MİMARİSİ DERSİ**

**PROJE ÖDEVİ**

**Sevgi Nur ÖKSÜZ  
21360859073**

**2025 Yılı**

**Hamming SEC-DED Kod Simülatörü Raporu**

**1. Giriş**

Bu rapor, bellek üzerindeki tek bit hatalarını düzeltebilen ve çift hataları tespit edebilen bir simülasyon arayüzü olan Hamming SEC-DED Code Simulator projesini tanıtmaktadır. Proje, Hamming kodlarının teorik altyapısını uygulamalı olarak göstererek kullanıcıların SEC-DED (Single Error Correction, Double Error Detection) sürecini adım adım deneyimlemesini sağlar.

**2. Amaçlar**

* SEC-DED Algoritmasını Görselleştirmek: Hamming kodu oluşturma, sendrom hesaplama, hata düzeltme adımlarını interaktif bir arayüzde sunmak.
* Kullanıcı Dostu UI: 8, 16 veya 32 bitlik girdilerle çalışabilen, anlaşılır buton ve uyarı mesajlarıyla desteklenen tasarım.
* Bit Görselleştirme: Hata eklenen ve düzeltilen bitleri renkli kutucuklarla vurgulayarak görselliği artırmak.
* Eğitsel Değer: Kullanıcıların Hamming kodlarının arka plandaki matematiksel işleyişini kavramalarına yardımcı olmak.

**3. Sistem Tasarımı**

**3.1 Mimari**

* HTML/CSS: Anlaşılır form yapısı, butonlar ve çıktı alanları.
* JavaScript (script.js):
  + calculateHamming(dataBits): Girdi uzunluğuna göre parite bitlerini belirler ve SEC-DED Hamming kodunu üretir.
  + syndromeAndCorrect(arr): Corrupted veride sendromu hesaplayarak tek bit hatasını düzeltir.
  + UI event handler’lar (genBtn, injectBtn, showBitsBtn) ile akış kontrolü.
  + renderGrid(flipPos, synPos): Dinamik olarak grid’i güncelleyip hatalı ve düzeltilen bitleri vurgular.
* styles.css: Responsive ve kontrastlı renk paleti, ızgara yatay kaydırma, seçili ve hata hücreleri için stil tanımları.

**3.2 Veri Akışı**

1. Kullanıcı veri uzunluğunu seçer ve ikili diziyi girer.
2. Generate Hamming Code butonuna basıldığında:
   * calculateHamming çağrılır.
   * Oluşturulan Hamming kodu encoded dizisine atanır.
   * Kod ekranda gösterilir.
3. Kullanıcı hata pozisyonu girip Inject Error and Correct butonuna basar:
   * Belirtilen pozisyondaki bit tersine çevrilir.
   * syndromeAndCorrect ile sendrom hesaplanır, hata düzeltilir.
   * Güncel kod ve sendrom ekranda sunulur.
4. Show Bit Visualization ile tablo görselleştirilir:
   * Üst satır bit numaralarını gösterir.
   * Alt satır bit değerlerini.
   * Hata hücresi sarı, düzeltilen hücre yeşil-mavi çerçeve ile işaretlenir.

**4. Kullanıcı Arayüzü**

* Select Data Length: 8/16/32 bit radyo düğmeleri.
* Input Field: Maksimum 32 karakter, sadece 0 ve 1 kabul eder.
* Buttons: Generate, Inject & Correct, Show Visualization.
* Output Areas: Monospace yazı tipiyle kod ve sonuç çıktı kutuları.
* Visualization: Kaydırılabilir grid, responsive.

**5. Test Süreci**

* 8-bit Senaryo: 10011001 → Generate → 00110010101 → hata pos=3 → Corrupted: ... → Corrected: ....
* 16-bit Senaryo: Rastgele 16 bit dizilerle doğru sendrom ve düzeltme testi.
* 32-bit Senaryo: Grid’in taşma yapmadan kaydırma ile gösterimi doğrulandı.
* Boundary Cases: Geçersiz giriş, yanlış bit pozisyonunda alert mesajları.

**6. Sonuç**

Bu proje, Hamming kodlarının SEC-DED yapısını adım adım interaktif bir ortamda sergileyerek hem teorik hem de uygulamalı eğitim imkanı sunmaktadır. UI/UX tasarımı ve görselleştirme teknikleri, kullanıcıların algoritmayı daha iyi kavramasını sağlamıştır.

**7. Adım Adım Uygulama ve Görseller**

Aşağıda, simülatörün her aşamasını gösteren ekran görüntüleri ile adım adım nasıl ilerlediği açıklanmıştır:

1. Veri Uzunluğu ve Girdi

* Radyo düğmelerinden 8, 16 veya 32 bit seçilir.
* Altındaki alana seçilen uzunlukta ikili veri girilir.

1. Hamming Kodu Üretimi

* Generate Hamming Code butonuna tıklanır.
* Ekranda hesaplanan Hamming kodu görüntülenir.

1. Hata Enjeksiyonu

* Flip bit at position alanına hata pozisyonu girilir (1 = en sağ bit).
* Inject Error and Correct butonuna tıklanır.
* Hata eklenmiş kod ve sendrom değeri çıktı olarak verilir.

1. Bit Görselleştirme

* Show Bit Visualization butonuna basılarak grid görünümü açılır.
* Üst satırda bit numaraları, alt satırda bit değerleri gösterilir.
* Hata pozisyonu sarı, sendromun işaret ettiği bit mavi çerçeve ile vurgulanır.

**metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.**

Şekil - 32 bitlik hamming code ve hata oluşturma çıktısı

**metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.**

Şekil - 16 bitlik hamming code ve hata oluşturma çıktısı

**metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.**

Şekil - 8 bitlik hamming code ve hata oluşturma çıktısı

***Sevgi Nur Öksüz, 2025***

**Proje Github Bağlantısı:** [**https://github.com/sevginuroksuz/hamming-code-simulator**](https://github.com/sevginuroksuz/hamming-code-simulator)

**Proje Youtube Bağlantısı:** [**https://www.youtube.com/watch?v=r8VfNTvKFow**](https://www.youtube.com/watch?v=r8VfNTvKFow)