Çevreci Gömülü Sistemler için Enerji Optimizasyonu: Algoritmaların Zaman Karmaşıklığı ile Güç Tüketimleri Arasındaki İlişkinin İncelenmesi Energy Optimization for Green Embedded Systems: Investigation of the Relationship Between Time Complexity and Power Usage

Buğra K. ACAR, Furkan ŞİMŞEKLİ, İlker AVCI
Bilgisayar Mühendisliği Bölümü
Hacettepe Üniversitesi
Ankara, Türkiye
{bugra_acar, furkansimsekli, ilkeravci}@hacettepe.edu.tr

***Özetçe* —Güç tüketimi, gömülü/IoT cihazlar için büyük önem arz etmektedir. Geçmişten günümüze bu cihazlar için algoritmalar tercih edilirken teorik zaman karmaşıklığı baz alınmıştır. Peki bu ne kadar isabetlidir? Bu araştırmada Arduino Uno R3 ve R4 mikrodenetleyicileri üzerinde Bubble, Insertion, Merge, Quick, Heap, Gnome, Radix, Shell, Comb ve Pancake Sort algoritmaları çalıştırılmış ve Nordic Power Profiler Kit II kullanılarak anlık çekilen akım ölçülmüştür. Çekilen akım ile verilen gerilim kullanılarak güç tüketimleri hesaplanmıştır. Güç tüketimleri ile zaman karmaşıklıkları arasında korelasyon analizi yapılmış ve [0.98, 1.0] arasında sonuçlar bulunmuştur ($p < 0.0001$). Bu bulgular sonucunda zaman kullanımı için optimize etmenin güç tüketimini de optimize ettiği kanıtlanmıştır. Farklı büyük dil modellerinin yazdığı kodların da güç tüketim analizleri yapılmış ve bazı öncül sonuçlar elde edilmiştir.**

Anahtar Kelimeler—gömülü sistemler, mikrodenetleyici mimarisi, algoritmalar, verimlilik, nesnelerin interneti, IoT, güç tüketimi, sıralama algoritmaları, çevreci hesaplama

***Abstract*—Power usage is a crucial point for embedded/IoT devices. Historically, theoretical time complexities were utilized during the choice of algorithm for these devices. But how accurate is this? In this research; Bubble, Insertion, Merge, Quick, Heap, Gnome, Radix, Shell, Comb and Pancake Sort algorithms have been ran on Arduino Uno R3 and R4 and current draws have been measured. Combining this with voltage, total power consumption has been calculated. The results show a [0.98, 1.0] correlation between the theoretical time complexities and power consumptions ($p < 0.0001$). The results indicate that optimizing for time spent also optimizes for power consumption. Comparison of power consumption of code generated by different large language models have also been analyzed and some preliminary results have been found.**

Keywords—embedded systems, microcontroller architecture, algorithms, efficiency, internet of things, IoT, power consumption, sorting algorithms, green computing

979-8-3315-6655-5/25/$31.00 ©2025 IEEE

I. GİRİŞ

Gömülü ve Nesnelerin İnterneti (IoT) cihazları, sinyal işleme ve iletişim uygulamaları başta olmak üzere, günümüz toplumunda pek çok alanda kritik rol oynamaktadır. Çamaşır makinelerinden insansız hava araçlarına kadar, bu cihazlar 21. yüzyılın yıldızları haline gelmiştir. 2025 itibarıyla 18,8 milyar cihaz bulunmakta olup, bu sayının 2030 yılına kadar 40 milyara ulaşması öngörülmektedir [2]. Bu tür cihazlar genellikle ya batarya ile çalışmakta ya da güneş enerjisi gibi kısıtlı bir güç kaynağına bağımlı olmaktadır. Dolayısıyla, bu tür cihazların tasarımında en önemli hususlardan biri enerji tüketiminin yönetilmesidir. Başarılı bir gömülü ürün geliştirmek, sistem performansını en üst düzeye çıkarırken enerji kullanımını en aza indirmeyi gerektirir [3].

Genellikle programların verimliliği, teorik zaman karmaşıklığına ve bellek kısıtlı ortamlarda kullanılan algoritmaların alan karmaşıklığına dayanarak değerlendirilir. Bu alandaki geçmiş araştırmalar çoğunlukla buyruk (instruction) düzeyinde analizlere ve simülasyon tabanlı modellere odaklanmıştır. Bu modeller, eski ve daha kısıtlı mikroişlemciler üzerinde tatmin edici doğruluk seviyelerine ulaşabilse de, modern mimariler için geliştirilmeye açık olduğu yazarlar tarafından düşünülmektedir. Çünkü modern mimarilerde kodun çalıştırılması; buyruk boru hattı, dallanma tahmini ve önbellek gibi birçok değişkene bağlıdır. Bu nedenle, bu çalışmada daha kanıta dayalı bir yaklaşım benimsenerek modern yazılımların yapı taşlarını oluşturan kapsamlı bir altyordam seti değerlendirilmeye karar verilmiştir. Elde edilen bulgularla, gömülü yazılım geliştirme sürecinde en uygun algoritmaların seçimine rehberlik edebilecek, ölçüme dayalı bir bilgi deposu oluşturmayı amaçlanmıştır. Çalışmamızda sunulan farklı altyordamların gerçek hayattaki enerji tüketimine dair içgörülerin, daha çevreci bir gelecek ve daha yüksek işlem kapasitesine sahip cihazlar için öncü olacağına inanılmaktadır.

II. İLGİLİ ÇALIŞMALAR

Bu alandaki araştırmalar, çeşitli platformlarda sıralama algoritmalarının enerji tüketimlerini ve zaman karmaşıklıklarıyla ilişkilerini değerlendirmiştir. Örneğin, Bunse vd., gömülü ve mobil ortamlarda farklı sıralama algoritmalarının enerji tüketimini karşılaştırmış ve zaman karmaşıklığı ile enerji tüketimi arasında doğrudan bir korelasyon olmadığını belirtmiştir [4]. Schuler ve Anderst-Kotsis, Android cihazlarda 12 farklı sıralama algoritmasının enerji tüketimini ve veri tipi seçiminin önemini araştırmıştır [5]. Verma ve Chowdhary ise akıllı telefonlarda çeşitli sıralama algoritmalarının enerji tüketimini ölçmüş, Quick Sort'u en verimli, Bubble Sort'u ise en çok enerji tüketen olarak belirlemiş ve "Enerji Karmaşıklığı" kavramını ortaya atmıştır [6].

Bu çalışmada ise Arduino Uno R3 ve R4 üzerinde geniş bir sıralama algoritması yelpazesinin güç tüketimleri yeterince hassas bir dijital ampermetre ile ölçülmüştür. Önceki çalışmalardan farklı olarak, teorik zaman karmaşıklığı ile gerçek güç tüketimi arasında çok güçlü bir pozitif korelasyon ($p < 0.0001$) tespit edilmiştir. Bu bulgu, Bunse vd.'nin aksine, zaman optimizasyonunun güç tüketimini de optimize ettiğini göstererek gömülü sistem geliştiricilerine pratik bir rehber sunmaktadır. Roy vd. tarafından önerilen enerji karmaşıklığı modeli, bulgularımız için teorik bir zemin oluşturabilir [7]. Bu çalışmanın, 8-bit AVR®ve 32-bit ARM®v7E-M mikrodenetleyici mimarilerinde elde edilen kanıta dayalı sonuçlarla literatüre katkı sağlayacağı düşünülmektedir.

Ayrıca, bu çalışmada gömülü sistemler için büyük dil modelleri (LLM) tarafından üretilen kodların enerji verimliliği değerlendirilmekte ve algoritma optimizasyonu bağlamında daha önce kapsamlı olarak incelenmemiş yenilikçi bir yaklaşım öne sürülmektedir.

III. YÖNTEM

A. Test Cihazları

TABLO I: TEST CİHAZLARI VE ÖZELLİKLERİ

Cihaz	Mikrodenetleyici	Mikroişlemci Mimarisi	Saat Hızı
R3	ATmega328P	Atmel AVR®(8-bit)	16MHz
R4	Renesas RA4M1	ARM®v7E-M (32-bit)	48MHz'e kadar

Gömülü sistemlerde enerji tüketimini ve mikrodenetleyici mimarisini incelemek için iki Arduino platformu kullanılmıştır. İlk cihaz olan Arduino UNO R3 SMD, makale boyunca R3 olarak adlandırılacaktır, 16 MHz frekansında çalışan 8 bitlik ATmega328P mikrodenetleyicisini barındırmaktadır. Program depolama için 32 KB flaş belleğe, çalışma zamanı verileri için 2 KB SRAM'e ve kalıcı veri depolama için 1 KB EEPROM'a sahiptir. R3, çeşitli açılardan sınırlı olmasına rağmen, karmaşık hesaplama gerektirmeyen senaryolarda uygun maliyetli bir alternatif olarak kullanılabilmektedir. Ayrıca; boşta (idle), bekleme (standby) ve güç kapatma gibi düşük güç modlarını destekleyerek, boşta kaldığı süre boyunca enerji tüketimini azaltma imkanı sunmaktadır [8].

İkinci cihaz olan Arduino UNO R4 Minima, makale boyunca R4 olarak adlandırılacaktır, 32 bit ARM®v7E-M mimarisine dayalı Renesas RA4M1 mikrodenetleyicisini kullanmaktadır. 48 MHz frekansında çalışabilen RA4M1, daha yüksek işlem gücü ve geliştirilmiş enerji verimliliği sunmaktadır. Program kodu için 256 KB flaş bellek, çalışma zamanı verileri için 32 KB SRAM ve kalıcı veri depolama için 8 KB EEPROM içermektedir. Bu iki platformun karşılaştırılmasıyla, bellek boyutunun ve mikrodenetleyici mimarisinin enerji tüketimi üzerindeki etkisi analiz edilebilir. R4, daha güçlü donanımıyla enerji tüketimi açısından daha verimlidir ve veri-yoğun işlemler için daha fazla yetenek sunmaktadır [9].

B. Kıyaslamalar

1) Sıralama Algoritmaları: Sıralama algoritmaları, özellikle verimlilik ve kaynak kısıtlamalarının büyük önem taşıdığı gömülü sistemlerde, çok çeşitli uygulamalarda kritik bir rol oynamaktadır. Gömülü sistemlerde sınırlı olan işlem gücü ve bellek kapasitesi nedeniyle, en uygun sıralama algoritmasının seçilmesi genel performans üzerinde önemli bir etkiye sahiptir. Bu kıyaslamada, 10 popüler sıralama algoritması her iki cihazda da test edilmiştir: Bubble Sort, Insertion Sort, Merge Sort, Quick Sort, Heap Sort, Gnome Sort, Radix Sort, Shell Sort, Comb Sort ve Pancake Sort.

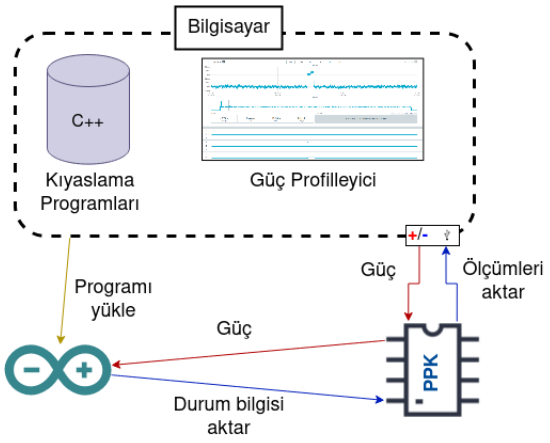
2) Kriptografik Algoritmalar: Gömülü sistemlerde hassas verilerin korunması büyük önem taşıdığı için, kriptografik algoritmalar veri güvenliğini sağlamak açısından kritik bir rol oynamaktadır. Bu kıyaslamada, farklı ChaCha şifreleme ve şifre çözme turları (8, 12 ve 20 tur), farklı anahtar uzunluklarında AES şifreleme ve şifre çözümü (128, 192 ve 256 bit) test edilerek çeşitli kriptografik algoritmaların performansı değerlendirilmiştir. Ayrıca AES (128/192/256)-GCM, ChaCha20-Poly1305 ve Acorn128 ile Ascon128 AEAD (ilişkili veri ile kimlik doğrulamalı şifreleme) şemaları da incelenmiştir.

3) Büyük Dil Modelleri Tarafından Üretilen Sıralama Algoritmaları: Geleneksel sıralama algoritmalarına ek olarak, büyük dil modellerinin sıralama kodu üretme yetkinliği de incelenmiştir. ChatGPT 4o, Gemini 1.5 Flash ve Claude 3.5 Haiku modellerine, 10 farklı sıralama algoritması için aynı şekilde yapılandırılmış istemler (prompt) verilmiş ve yalnızca sıralama fonksiyonu ile açıklayıcı kısa dokümantasyon üretmeleri istenmiştir [1]. Bu sabit yapı, deneyin tekrarlanabilirliğini güvence altına alarak modellerin gömülü sistemler için uygunluklarını adil biçimde karşılaştırmayı amaçlamıştır.

4) Diğer Algoritmalar: Yaygın kullanılan bazı algoritmalar diğer algoritmalar başlığı altında toplanmıştır. LCG (Doğrusal Artıklık Üreteci) ve MT19937 (Mersenne Twister) rastgele sayı üreteçleri, doğrusal ve ikili arama, matris çarpımı, ikili GCD (En Büyük Ortak Bölen), Dijkstra, Karatsuba çarpımı, fibonacci ve faktöriyel algoritmalarının yinelemeli (iterative) ve özyinelemeli (recursive) versiyonları ve temel bir toplama işlemi, programlarda yaygın olarak kullanılan algoritmaların çeşitliliğini temsil edecek şekilde seçilmiştir.

C. Deney ve Ölçüm Düzeni

Tüm deneylerde, Arduino kartlarından çekilen akımı gerçek zamanlı olarak ölçmek için Nordic® Power Profiler Kit II (makale boyunca PPK olarak anılacaktır) kullanılmıştır. PPK, 100.000 ölçüm/saniye örnekleme hızına, ±200nA hassasiyete ve 1A'e kadar akım ölçme kapasitesine sahiptir [10]. Deneylerde PPK kaynak ölçer (sourcemeter) modunda kullanılmış ve Arduino kartlarını 5V'a kadar besleyebilmesi nedeniyle güç kaynağı olarak görev yapmıştır.



Şekil 1: Deney düzeneği diyagramı

Şekil 1'de gösterildiği gibi, deneyler PPK'yi Arduino kartlarına bağlayarak gerçekleştirilmiştir ve bütün devrenin güç kaynağı olarak bilgisayar kullanılmıştır. PPK'nin uyarlanabilir kaynak modu sayesinde ise test cihazları deney türüne göre 3,3V ya da 5V pini üzerinden beslenilmiştir. Deneyi oluşturan algoritmalar için test cihazlarıyla uyumlu kıyas programı yazıldıktan sonra aşağıdaki akışa göre ölçümler alınmıştır.

1: Ölçüm cihazı (PPK) bilgisayara bağlanır
2: **Her bir** test cihazı **için yap**
3: **Her bir** kıyas programı **için yap**
4: Test cihazı, bilgisayara bağlanır
5: Kıyas programı test cihazına yüklenir
6: Test cihazının bilgisayarla bağlantısı kesilir
7: Test cihazı ölçüm cihazına bağlanır
8: Ölçüm uygulamasında deney başlatılır
9: Deney sonlandığında sonuçlar kaydedilir
10: Test cihazının ölçüm cihazıyla bağlantısı kesilir
11: **bitti**
12: **bitti**
13: Elde edilen ölçüm sonuçları analiz edilir

IV. Deney Bulguları

TABLO II incelendiğinde, sıralama algoritmalarına arasında Quick Sort algoritmasının hem en hızlı hem de enerji verimliliği en yüksek algoritma olduğunu görülmüştür. TABLO III'e bakıldığında, algoritmaların teorik zaman karmaşıklığı ile yüksek korelasyonda olduğunu görülmektedir. TABLO IV'teki istatistiksel anlamlılık verilerine baktığımızda da P değerinin ilgili algoritmaya ait teorik zaman karmaşıklığında en düşük olduğu görülebilmektedir. Sonuçlara bakıldığında, Radix Sort'un zaman karmaşıklığı Quick Sort'tan daha iyi olsa da, $O(n)$ - $O(n \log n)$, Radix Sort'un sabit katsayısının büyüklüğünden dolayı gömülü cihazlarda işlenebilecek veri miktarları için pratikte önemli olmadığını görülmüştür.

Ayrıca, Radix Sort R3'te ciddi şekilde düşük performans göstermiştir. R3, donanım tabanlı tam sayı bölmesi buyruklarına sahip olmadığı için Radix Sort'taki bölme işlemlerinin yazılım emülasyonu ile yapıldığına; bunun ise yavaşlamaya ve fazladan enerji tüketimine neden olduğu sonucuna varılmıştır. Bu istisna dışında, enerji tüketimi ile zaman karmaşıklığı arasında beklenen düzeyde bir uyum olduğu saptanmıştır.

TABLO II: Sıralama Algoritmalarının R3 ve R4 Üzerinde Karşılaştırılması

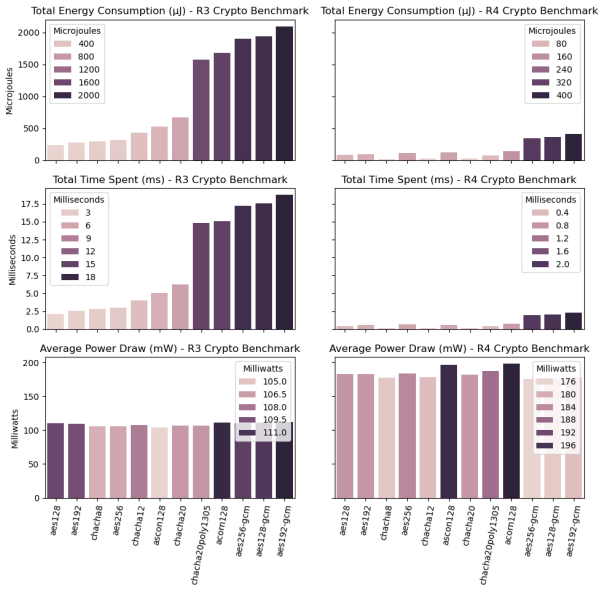
Algoritma†	Süre (ms)		Ortalama Güç (mW)		Toplam Enerji (μJ)	
	R3	R4	R3	R4	R3	R4
Quick Sort	2,32	0,30	119,006	177,647	276	54
Shell Sort	3,20	0,48	119,404	171,806	382	83
Comb Sort	3,71	0,47	119,408	172,915	443	81
Heap Sort	4,68	0,56	119,850	178,518	561	100
Merge Sort	5,24	0,72	118,315	172,915	620	124
Insertion Sort	5,44	0,85	119,675	178,468	651	151
Gnome Sort	18,70	2,53	120,190	173,785	2247	440
Bubble Sort	20,80	1,89	118,656	179,626	2468	339
Pancake Sort	23,30	3,25	119,660	178,208	2788	579
Radix Sort	24,48	1,02	116,701	160,922	2857	164

† R3'te 16 bit, R4'te ise 32 bit büyüklüğe sahip olan 'int' türünde, 128 elemanlı, önceden oluşturulmuş rastgele tam sayı dizisi üzerinde test edilmiştir.

TABLO III: Sıralama Algoritmalarının R4'teki Performansının Zaman Karmaşıklıkları ile Korelasyonu

Algoritma†	$O(\log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$
Quick Sort	0,85	1,00	1,00	0,98
Shell Sort	0,82	0,99	1,00	0,99
Comb Sort	0,84	1,00	1,00	0,98
Heap Sort	0,86	1,00	1,00	0,98
Merge Sort	0,86	1,00	1,00	0,98
Insertion Sort	0,74	0,97	0,98	1,00
Gnome Sort	0,74	0,97	0,98	1,00
Bubble Sort	0,75	0,97	0,98	1,00
Pancake Sort	0,75	0,97	0,98	1,00
Radix Sort	0,88	1,00	1,00	0,97

† 8, 16, 32, 64, 128, 256, 512 ve 1024 elemanlı tam sayı listeleri üzerinde koşulmuştur.



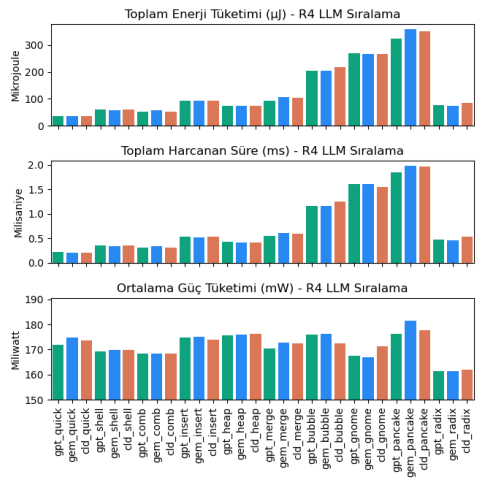
Şekil 2: Kriptografi algoritmalarının R3 ve R4'teki performansları

Şekil 2'deki kriptografik algoritmalarla ilgili olarak, sonuçlar cihaza bağlı olarak değişmektedir. R3'te primitif algoritmalar arasında AES'in fakat AEAD'lar arasında Ascon128'in en verimli olduğu tespit edilmiştir. Pratikte AEAD'lar doğrulamalı olduklarından dolayı daha yararlı ol-

TABLO IV: SIRALAMA ALGORİTMALARININ R4'TEKİ PERFORMANSININ ZAMAN KARMAŞIKLIKLARI İLE KORELASYONLARININ İSTATİSTİKSEL ANLAMLILIĞI

Algoritma	$O(\log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$
Quick Sort	0,015	4,4e-7	2,3e-9	9,3e-5
Shell Sort	0,024	1,1e-5	8,4e-7	1,4e-5
Comb Sort	0,017	7,3e-7	1,4e-9	8,5e-5
Heap Sort	0,014	8,8e-8	2,2e-12	1,5e-4
Merge Sort	0,013	3e-8	2,2e-10	1,8e-4
Insertion Sort	0,055	4,2e-4	1,4e-4	2,5e-11
Gnome Sort	0,055	4,3e-4	1,5e-4	3,9e-11
Bubble Sort	0,054	3,9e-4	1,3e-4	2e-16
Pancake Sort	0,054	3,8e-4	1,3e-4	1,4e-15
Radix Sort	0,0093	5,7e-16	1,5e-7	3,9e-4

dukları [11] için AES'in AEAD modları yerine Ascon128'in kullanılmasının daha uygun olduğu görülmektedir. R4'te ChaCha20Poly1305'in genel olarak en verimli olduğu, ChaCha yordamlarının ise AES ve Acorn128/Ascon128'den daha verimli olduğu görülmüştür. AES altyordamlarının tamamen yazılım tabanlı bir implementasyon olması ve donanım hızlandırma kullanılmaması burada önemli bir husustur. Şekil 3'teki LLM karşılaştırılmalarında ChatGPT'nin ortalama olarak Gemini'dan %3,09 daha verimli, Claude'dan ise %3,67 daha verimli kod ürettiği, ancak genellikle birbirlerine çok yakın sonuçlar verdiği görülmüştür.



Şekil 3: Büyük dil modellerinin R4 için yazdığı algoritmaların performansları

R3'ün performansta bir düşüş yaşamadan 2,735V'luk bir güç kaynağı ile çalışabildiği, bunun da enerji kullanımını azalttığı görülmüştür. Ancak kararlılığı artırmak adına kartın daha yüksek bir gerilimle çalıştırılması önerilmektedir. 3,3V ve 5V deneyleri karşılaştırıldığında enerji kullanımında %62 oranında bir azalma tespit edilmiştir. Genel olarak, R4'ün hem daha hızlı hem de daha verimli olduğu tespit edilmiştir. Bu sonuç, R4'ün daha modern ve daha verimli mikroişlemci mimarisine sahip olmasına bağlanmıştır.

V. SONUÇ

Bu araştırma, algoritmaların güç tüketimlerini zaman karmaşıklıklarıyla ilişkilendirerek incelemiştir. Yapılan deneyler, enerji tüketimi ile algoritmaların zaman karmaşıklığı arasında doğrudan bir ilişki olduğunu göstermektedir. Özellikle sıralama algoritmalarının enerji verimliliği analiz edildiğinde, Quick Sort gibi zaman verimli algoritmaların enerji tüketimi açısından da avantajlı olduğu gözlemlenmiştir. Bununla birlikte, farklı platformlar üzerinde yapılan testler, Radix Sort gibi belirli algoritmaların güç tüketiminin, kullanılan donanımın özellikleriyle de yakından ilişkili olduğunu ortaya koymuştur. Zaman karmaşıklığına dayalı optimizasyonların, enerji verimliliği sağlamak adına önemli bir rol oynadığı, bu araştırma ile kanıtlanmıştır. Ayrıca büyük dil modelleri arasında yapılan analizde enerji verimliliği açısından %3'e kadar çıkan farklar gözlemlenmiştir.

Sonuç olarak, bu çalışma gömülü sistemler için enerji optimizasyonunun, algoritma seçiminden donanım mimarisine kadar birçok faktöre bağlı olduğunu ortaya koymuştur. Bu alandaki bulgular, hem teoriye dayalı hem de uygulamalı enerji verimliliği analizlerinin gelecekteki gömülü yazılım geliştirme süreçlerinde daha etkin bir şekilde kullanılabileceğini işaret etmektedir. Büyük dil modelleri üzerine olan bulgular gelecekteki bir araştırmada daha detaylı incelenecektir. Araştırma kapsamında alınan ölçümler, çalıştırılan kodlar ve detaylı deney bilgisi herkese açık bir GitHub deposuna [1] yüklenmiştir.

KAYNAKLAR

[1] https://github.com/FOSSketeers/embedded-power-analysis

[2] IoT Analytics GmbH, "Connected IoT device market update—Summer 2024," 2024. https://web.archive.org/web/20250222024327/https://iot-analytics.com/number-connected-iot-devices/ (erişildi Mar. 09, 2025).

[3] B. C. E. S. Nogueira et al., "Performance and Energy Consumption Evaluation of Embedded Applications: A Method Based on Platform's Behavioral Model," 21st International Symposium on Computer Architecture and High Performance Computing, pp. 135–142, Oct. 2009, doi: 10.1109/sbac-pad.2009.27.

[4] C. Bunse, H. Höpfner, E. Mansour, ve S. Roychoudhury, "Exploring the Energy Consumption of Data Sorting Algorithms in Embedded and Mobile Environments", 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware. IEEE, ss. 600-607, 2009. doi: 10.1109/mdm.2009.103.

[5] A. Schuler and G. Anderst-Kotsis, "Examining the Energy Impact of Sorting Algorithms on Android: An Empirical Study," MobiQuitous '19: Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, pp. 404–413, Nov. 2019, doi: 10.1145/3360774.3360808.

[6] M. Verma and K. R. Chowdhary, "Analysis of energy consumption of sorting algorithms on smartphones," SSRN Electronic Journal, Jan. 2018, doi: 10.2139/ssrn.3168550.

[7] S. Roy, A. Rudra, and A. Verma, "An energy complexity model for algorithms," ITCS 2013, Jan. 2013, doi: 10.1145/2422436.2422470.

[8] Microchip, "ATmega328P 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash DATASHEET," 2015. https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf (erişildi Mar. 09, 2025).

[9] Renesas Electronics, "RA4M1 Group Datasheet," Sep. 29, 2023. https://www.renesas.com/en/document/dst/ra4m1-group-datasheet (erişildi Mar. 09, 2025).

[10] Nordic Semiconductor ASA., "Power Profiler Kit II Technical Documentation," 2025. https://docs.nordicsemi.com/bundle/ug_ppk2/page/UG/ppk/PPK_user_guide_Intro.html (erişildi Mar. 09, 2025).

[11] N. Ferguson, B. Schneier, and T. Kohno, Cryptography Engineering: Design Principles and Practical Applications. Wiley, 2010, ss 220.