GAZI UNIVERSITY
FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING
CENG318 - MICROPROCESSORS

MIDTERM RESEARCH PROJECT

ARM64 BASED MICROPROCESSORS

LECT. PHD MUHAMMET ÜNAL

SELİN CANSU AKBAŞ
191180005

Submission Date

09.05.2023

MAY 2023

İÇİNDEKİLER

Sayfa

ŞEKİLLER LİSTESİ

Şekil **Sayfa**

TABLOLAR LİSTESİ

Tablo **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklamalar
ARM	Advanced RISC Machines
AArch64	ARM Architecture 64-bit
AES	Advanced Encryption Standard
ASIC	Application-Specific Integrated Circuit
ASSP	Application-Specific Standard Product
CCR	Condition Code Register
CPU	Central Processing Unit
DES	Data Encryption Standard
EEPROM	Electrically Erasable Programmable Read-Only Memory
ELR	Exception Link Register
FPGA	Field Programmable Gate Array
IoT	Internet of Things
ISA	Industry Standard Architecture
iOS	iPhone Operating System
LCD	Liquid Crystal Display
MMU	Bellek Yönetim Birimi
PC	Program Counter
PLC	Programmable Logic Controller
PMSA	Korumalı Bellek Sistemi Mimarisi
RAM	Random Access Memory
ROM	Read-Only Memory
SIMD	Single Instruction Multiple Data
SoC	System-on-Chip
SP	Stack Pointer

SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus
VMSA	Sanal Bellek Sistemi Mimarisi
VPS	Virtual Private Server

1. GİRİŞ

ARM64 tabanlı mikroişlemciler, mobil cihazlarda, bilgisayarlarda, sunucularda ve başka gömülü sistemlerde de yaygın olarak kullanılan yeni nesil işlemciler olarak adlandırılabilmektedirler. Bu mikroişlemcilerin 64-bit mimarisi bulunmaktadır. Diğer mikroişlemcilere göre daha güçlüdürler, daha fazla bellek desteği sunarlar ve düşük güç tükettikleri için enerji verimlidirler. ARM64 işlemciler, birden fazla çekirdekli yani multi-core olabilmektedirler. Aynı anda birçok veri işleyebilirler. ARM64 tabanlı olan cihazlar çoğunlukla yüksek performanslı işlemler yapabilmek için kullanılırlar. Örnek vermek gerekirse veri işleme, yapay zekâ, grafik işleme gibi birçok alanda kullanılabilirler. ARM64 işlemciler, birçok register kullanarak daha hızlı ve verimli işlemler yapabilirler. Bu registerlar, işlemcinin çekirdekleri için gereken verileri ve komutları saklarlar. Başka bir tanımlama yapmak gerekirse verileri saklamak ve işlemek için kullanılan bir bellektir de denilebilir. Register kullanımı, aynı zamanda işlemci performansını da doğrudan etkiler. Yani, ARM64 tabanlı mikroişlemciler, yüksek performans, düşük güç tüketimi ve daha fazla bellek desteği sunan bir işlemci mimarisi olarak önümüze çıkarlar. Bu işlemcilerin tasarımları, farklı cihazlarda kullanabilecek şekilde yapılmıştır ve ideal bir seçenek olarak karşımıza çıkmaktadır. ARM64 tabanlı mikroişlemcilerin registerları da işlemcilerin hızının artmasını ve verimli çalışmasını sağlamasının yanı sıra ARM64 mikroişlemcilerin yüksek performanslı işlemler yapmasına da yardımcı olmaktadırlar.

2. ARM MİMARİSİNE GİRİŞ

2.1. ARM Nedir?

ARM, yani Advanced RISC Machines; mobil cihazlarda ve diğer gömülü sistemlerde yaygın olarak kullanılan bir tür işlemci mimarisidir [1]. ARM mimarisi, dünya çapında yaygın olarak kullanılan bir işlemci mimarisidir. Bu mimari, İndirgenmiş Komut Kümesi Bilgisayarı (RISC) ailesine aittir ve akıllı telefonlar, tabletler ve IoT cihazları gibi birçok bilgisayar işlemcisi için temel CPU olarak kullanılır. ARM mimarisi, 32 ve 64 bit sürümleri bulunur ve genellikle gömülü sistemler ve bozuk cihazlar gibi düşük güç tüketimine ihtiyaç duyduğu yerlerde tercih edilir. Bu mimarinin diğer RISC işletim işlemcilerine göre yüksek performans ve x86-x64 işlemcilere göre daha ekonomik olması da tercih edilme sebepleri arasındadır [2].

ARM işlemcileri, sadece tüketici elektroniği için değil aynı zamanda uzun pil ömrü ve yüksek işlem hızları gerektiren cihazlar için ideal olan düşük güç tüketimi ve yüksek performanslarıyla da bilinmektedir. Bu nedenle, güvenlik açısından kritik ve ultra-güvenilir uygulamalar dahil olmak üzere otomotiv ve koruyucu sistemler gibi birçok farklı durumda da kullanılmaktadırlar. ARM işlemcileri, öngörülebilirliği, güvenilirliği ve kullanım kolaylığı nedeniyle milyarlarca cihazda bulunmaktadır. Bu mimarinin düşük üretim maliyeti ve yüksek güç sağlamaları, ARM'nin popülerliğinin bir başka nedenidir [3].

ARM mimarisi aşağıda belirtilen unsurları desteklemektedir:

- AArch64; 64 bit olan yürütme durumu,
- AArch32; ARM mimarisinin geçmiş sürümleri ile uyumlu bir 32 bit olan yürütme durumu.

ARM mimarisi, tanıtıldığı günden bu yana büyük ilerlemeler kaydetmiş ve günümüzde de gelişmeye devam etmektedir. 1 ile 9 arasında numaralandırılmış olan dokuz ana mimari sürüm belirlenmiştir. Bu mimarilerin üç sürümü eskimesi sebebiyle güncel kullanımda yer almamaktadır.

AArch64 ve AArch32 terimleri, 64 bit ve 32 bit yürütme durumlarını belirtmek için kullanılmaktadır:

- AArch64, 64 bit olan yürütme durumunda adresler 64 bit registerlarda depolanmakta; bu süreçte temel komut setleri de 64 bitlik registerları kullanmaktadır. Buna ek olarak AArch64, A64 komut setini desteklemektedir.
- AArch32, 32 bit olan yürütme durumunda adresler 32 bit registerlarda depolanmakta; bu süreçte temel komut setleri de 32 bitlik registerları kullanmaktadır. Buna ek olarak AArch32, T32 ve A32 komut setlerini desteklemektedir [20].

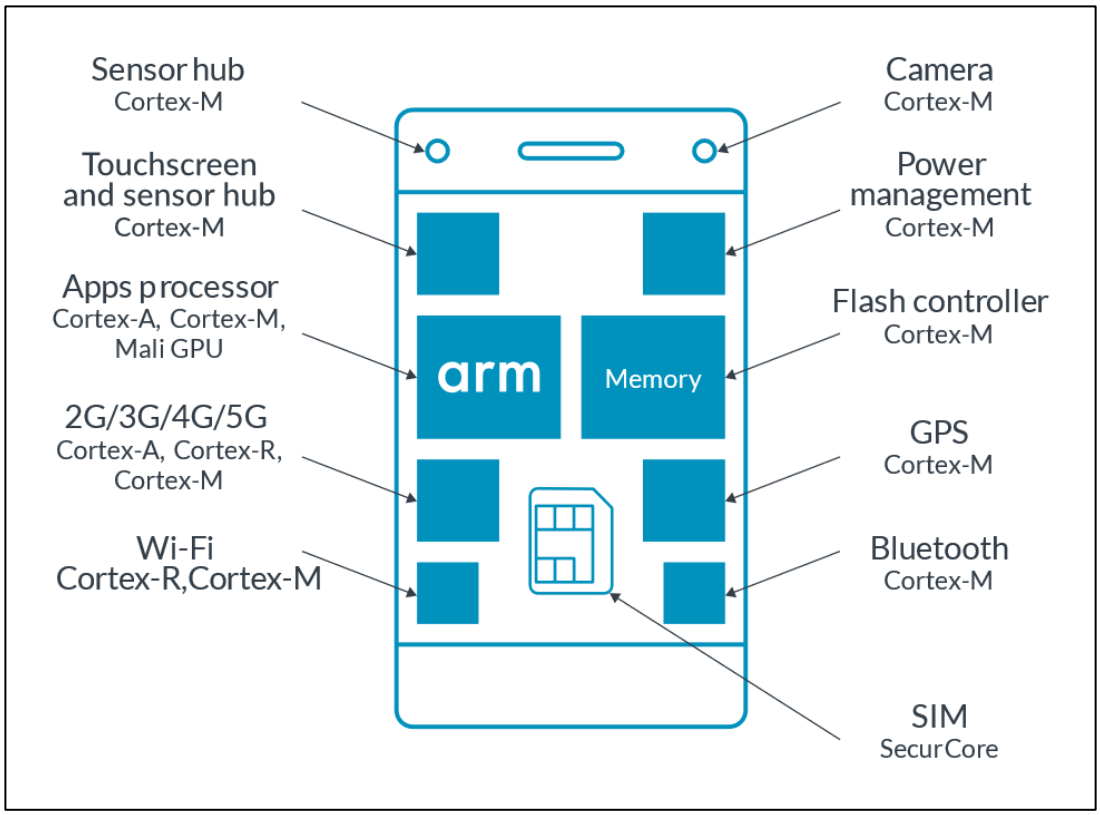
Şekil 2.1'deki örnekteki akıllı telefon aşağıdaki işlemci türlerini içermektedir:

- Ana CPU olarak A-profil işlemci, Android gibi zengin bir işletim sistemini çalıştırmaktadır.
- Hücresel modem, R-profil işlemcisine dayanmakta ve bağlantı sağlamaktadır.
- Sistem güç yönetimi gibi işlemleri gerçekleştiren birkaç M-profil işlemcisi bulunmaktadır.
- SIM kart, SecurCore adı verilen ek güvenlik özelliklerine sahip bir M-profil işlemcisi kullanmaktadır. SecurCore işlemciler, akıllı kartlarda yaygın olarak kullanılmaktadır.

ARM mimarisi, bir işlemcinin işlevselliğini belirleyen özellikleri ifade etmektedir. Mimariler, bir işlemcinin;

- Nasıl davranacağını,
- İçerdiği talimat setini ve
- Talimatların hangi işlemleri gerçekleştirdiğini tanımlamaktadırlar.

ARM mimarisi, donanım ve yazılım arasındaki bir sözleşme olarak nitelendirilebilmektedir. Mimari, donanımın sağlayacağı işlevleri açıklamakta ve yazılımın açıklanan bu işlevlere güvenebilmesini sağlamaktadır. Mimari ve mikro mimari, bazı özelliklerin isteğe bağlı olduğunu belirtmektedir.



Şekil 2.1. ARM Tabanlı Bir Sistem Örneği [9]

ARM mimarisi;

- Komut seti,
- Register seti,
- İstisna modeli,
- Bellek modeli,
- Hata ayıklama,
- İzleme ve
- Profil oluşturma terimlerini içermektedir.

Her biri belirli bir amaç için tasarlanmış üç adet profil bulunmaktadır (Tablo 2.1).

Core	**A**	**R**	**M**
Profile	Application	Real-time	Microcontroller
Application	Bir Bellek Yönetim Birimi'ne (MMU) dayalı bir Sanal Bellek Sistemi Mimarisini (VMSA) desteklemektedir.	Gerçek zamanlı gereksinimleri olan sistemlerde hedeflenmektedir. Ağ ekipmanlarında ve gömülü kontrol sistemlerinde yaygın olarak bulunmaktadır. Tıbbi cihazlarda, PLC'de, ECU'da, aviyonikte, robotikte bulunmaktadır.	Bir MMU'ya dayalı Korumalı Bellek Sistemi Mimarisini (PMSA) desteklemektedir. Güç yönetimi, G/Ç, dokunmatik ekran, akıllı pil ve sensör denetleyicileri için ASIC'lerde, ASSP'lerde, FPGA'larda ve SoC'lerde bulunmaktadır.
	Windows, Linux, Android veya iOS gibi bir işletim sistemini çalıştıran yüksek performanslı cihazlarda bulunmaktadır.	Düşük gecikme süresi ve yüksek düzeyde güvenlik gerektiğinde kullanılmaktadır. Örneğin, bir otomobildeki elektronik fren sistemi, otonom dronlar vb.	Küçük, yüksek güç verimliliğine sahip cihazlarda yer almaktadır. Birçok IoT cihazının merkezinde bulunmaktadır.
	A64, A32 ve T32 komut setlerini desteklemektedir.	A64, A32 ve T32 komut setlerini desteklemektedir.	T32 komut setinin bir çeşidini desteklemektedir.

Tablo 2.1. ARM Mimarisi Hakkında Detaylı Tablo [8]

2.2. ARM Mimarisinde Bulunan Bileşenler ve Fonksiyonları

ARM işlemcisi, tek bir komut seti mimarisi ve geriye dönük uyumluluk sağlayan programlama modellerine sahiptir. Bu işlemci, derleyici ve programcı tarafından erişilebilen bir register kümesine sahip olup, mimari register sayısının artırılmasını ve her register alanına bir bit eklenmesini gerektirmektedir [4]. ARM işlemcileri,

konuşma çözme (speech decoding), hedef izleme (target tracking) ve çoklu saatli gömülü sistemler (multi-clock embedded systems) gibi farklı uygulamalarda kullanılmaktadır [5].

2.3. ARM Mimarisi ve Bellek Yönetimi

Bellek yönetimi, bir bilgisayar mimarisinin temel bir özelliğidir ve ARM mimarisi, sanal bellek desteği sağlayan bir Bellek Yönetim Birimi'ne (MMU) sahiptir. MMU, sanal bellek desteği sağlamakta ve sanal adresleri fiziksel adreslerle eşleştirerek bellek bölgelerine erişimi kontrol etmektedir. Bu sayede bellek koruması sağlanmakta ve yazılım hatalarının neden olduğu bellek çarpışmaları önlenebilmektedir. Bunlara ek olarak, ARM işlemcileri, farklı bellek türlerini de desteklemektedir: ROM, RAM ve flash bellek. Bu bellek türleri, farklı bellek gereksinimlerini karşılamak için kullanılabilmektedir. Özellikle gömülü sistemlerde ARM işlemcileri sıklıkla kullanılmakta ve bu sistemlerde farklı bellek türlerine ihtiyaç duyulabilmektedir.

2.4. ARM Mimarisi ve Gömülü Sistemler

ARM, mobil cihazlar ve diğer birçok üst düzey gömülü sistem için baskın işlemci mimarisidir. ARM işlemcileri genellikle mikrodenetleyiciler, sahada programlanabilir geçit dizileri (FPGA'lar) veya çip üzerinde sistemler olarak uygulanır. ARM, değiştirilmemiş konuk işletim sistemi ikili dosyalarının yürütülmesine izin verecek olan sanallaştırma için mimari desteğini duyurdu. Bu, sanallaştırma desteği gerektiren gömülü sistemler için faydalı olacaktır [1].

ARM, mobil cihazlar ve birçok üst düzey gömülü sistemler için baskın işlemci mimarisi olarak nitelendirilebilir. ARM işlemcileri genellikle mikrodenetleyiciler, sahada programlanabilir geçit dizileri (FPGA'lar) veya çip üzerinde sistemler olarak uygulanmaktadır. Bu işlemciler, yüksek performanslı bir işlem gerektiren akıllı telefonlar, tabletler ve diğer mobil cihazlar için de kullanılmaktadır. ARM mimarisi, birçok uygulama için düşük güç tüketimi ve yüksek performans avantajları sağlamaktadır.

ARM mimarisi, yüksek derecede programlanabilir bir işlemci olması nedeniyle geniş bir kullanım alanı sunmaktadır. Bu işlemciler, değiştirilmemiş konuk işletim sistemi ikili dosyalarının yürütülmesine izin veren sanallaştırma desteği de sağlamaktadır. Bu, özellikle gömülü sistemlerde kullanışlı olmaktadır. ARM işlemcileri ayrıca, yüksek performans ve düşük güç tüketimi için optimize edilmiş çok sayıda özel birim içermekte, bu da ARM tabanlı cihazların verimliliğini arttırmaktadır [1].

ARM mimarisini kullanan gömülü sistemlerin pek çok farklı uygulaması bulunmaktadır. Bu uygulamalara örnek olarak, "ARM Cortex-M4 İşlemci ve Omnivision OV7670 Renkli Kamera Kullanılarak Tasarlanan Gerçek Zamanlı Meyve Algılama Sistemi" verilebilir [6]. Bu sistem, robotik bir kol yardımıyla meyveyi ağaçtan otomatik olarak seçmek ve toplamak amacıyla tasarlanmıştır. Başka bir örnek ise, gömülü bir ARM işlemci üzerinde çalışan bir Polar Codes kod çözücü yazılım uygulamasıdır. Bu kod çözücü, ARM işlemcilerinin NEON komut setini kullanarak algoritmanın paralelliğinden yararlanmakta ve kod çözücünün verimliliğini artırmaktadır [7]. Bu örnekler, ARM mimarisi sayesinde güçlü ve yüksek performanslı gömülü sistemlerin tasarlanabileceğini göstermektedir.

2.5. ARM Mimarisinde Bulunan I/O Arabirimleri

Bellek yönetimi yanı sıra G/Ç arabirimleri de ARM mimarisinin önemli bir yönüdür. ARM işlemcileri, UART, SPI, I2C ve USB gibi çeşitli G/Ç arabirimleri sağlamaktadır. Bu arabirimler, harici cihazlarla, örneğin sensörler, aktüatörler ve bellek cihazlarıyla iletişim kurmak için kullanılmaktadırlar.

Gömülü sistemlerde hata ayıklama ve iletişim kurma amacıyla kullanılan basit bir seri iletişim arabirimi olan UART, ARM mimarisi tarafından desteklenen çeşitli G/Ç arabirimlerinden biridir. SPI arayüzü, flash bellek, sensörler ve LCD ekranlar gibi cihazlarla iletişim kurmak için kullanılan senkronize bir seri iletişim arayüzüdür. I2C arayüzü, EEPROM'lar, sıcaklık sensörleri ve gerçek zamanlı saatler gibi cihazlarla iletişim kurmak için kullanılan iki telli bir seri iletişim arayüzüdür. USB arabirimi ise klavye, fare ve flash sürücü gibi aygıtları sisteme bağlamak için kullanılan yüksek hızlı

bir seri iletişim arabirimidir. Bu G/Ç arabirimleri, ARM mimarisi tarafından sağlanan çeşitli iletişim protokollerine uygun olarak çalışır ve harici cihazların kontrolüne olanak tanımakta; bu özellikleri ile, ARM işlemcilerinin geniş bir uygulama yelpazesinde kullanılmasını sağlamaktadır.

3. ARM64 MİMARİSİ

ARM64 mimarisi, mikroişlemcilerde kullanılan bir mimari türüdür [10]. 64 bitlik bir mimari olan ARM64, çeşitli uygulamalarda kullanılmaktadır. Özellikle enerji verimli hesaplama kimyası, güvenlik işlemcileri ve gömülü uç bilgi işlem gibi alanlarda sıkça kullanılmaktadır. ARM64, ARM mimarisi ailesinin 64-bitlik bir uzantısıdır ve özellikle enerji verimliliği ve düşük gecikme süresi gibi tasarım ilkeleri ile öne çıkmaktadır. ARM tabanlı mikroişlemcilerde kullanılan bu mimari, büyük register dosyaları, register yeniden adlandırma, öngörü, spekülasyon ve yazılım pipelining için özel destek gibi özelliklere sahiptir.

ARM64 mimarisi, düşük enerji tüketimi ve yüksek performansı bir arada sunan bir mimaridir. Bu özellikleri sayesinde düşük enerji tüketen ARM64 çekirdekleri ve FPGA hızlandırıcılarına dayalı exascale hazır bir bilgi işlem platformu prototipi tasarlamayı ve geliştirmeyi amaçlayan ExaNeSt ve EuroExa H2020 AB tarafından finanse edilen projeler gibi yüksek performanslı bilgi işlem sistemlerinde de kullanılmaktadır [11]. Ayrıca, ARM64 mimarisi, bulut tabanlı uygulamalar ve özellikle büyük veri iş yükleri için gelecek vaat eden bir mimaridir.

Tüm bunların yanı sıra, ARM64 mimarisi güvenlik konusunda da öne çıkmaktadır. ARM64 tabanlı sistemler, donanım tabanlı kök güvenlik modülleri (HSM) ile birlikte kullanıldığında, özellikle kriptografi işlemleri için güvenli bir ortam sağlama olasılığına sahip olmaktadır. Bu sayede, finansal işlemler, veri işleme ve depolama gibi hassas verilerin işlenmesinde kullanılan sistemlerde güvenli bir yapı oluşturulabilmektedir [12].

ARMv7-A ve ARMv8-A, ARM mimarisinin farklı nesillerini temsil eden iki işlemci mimarisi olup, özellikle çoklu çekirdekli sistemlerde kullanılmaktadırlar. ARMv7-A, 2007 yılında piyasaya sürülmüş olan geçmiş nesil ARM işlemci mimarisidir. Diğer yandan, ARMv8-A 2011 yılında tanıtılmış olup, daha yüksek performans, ölçeklenebilirlik ve güvenlik özellikleri ile öne çıkmıştır. ARMv8-A, daha yeni bir mimari olduğundan çoğu modern sistemde kullanılmaktadır.

ARMv8-A mimarisi, çoklu çekirdekli işlemcilerin hızlı ve verimli bir şekilde çalışmasını sağlamak amacıyla çoklu kopyalı atomik bir bellek modeline sahip şekilde tasarlanmıştır. Bu, yazma işlemlerinin aynı anda tüm iş parçacıkları tarafından görülebileceği anlamına gelmektedir. Diğer yandan, ARMv7-A mimarisi, çoklu kopyalı olmayan bir atomik bellek modeline sahiptir. Bu da yazma işlemlerinin bazı ileti dizileri tarafından görünür hale gelmeden önce tüm ileti dizileri tarafından görülebilmesi anlamına gelmektedir.

ARMv8-A mimarisi, özellikle güvenliği arttırmak için bazı özellikler içermektedir. Bu özellikler, işlemci çekirdekleri arasındaki bellek paylaşımını ve korumayı geliştirmiştir. Ayrıca, bu mimari, SIMD (Single Instruction Multiple Data) işlemleri için genişletilmiş destek sunmaktadır. SIMD işlemleri, aynı anda birden fazla veri işleme işlemi olup, özellikle görüntü, video ve ses gibi veri yoğun uygulamalar için faydalı olarak nitelendirilmektedir [14].

ARMv7-A ve ARMv8-A mimarilerinin de içinde bulunduğu ARM64 komut seti mimarisi (ISA), farklı türde hesaplama ve sistem gereksinimlerini desteklemek için tasarlanmış bir dizi komut seti içermektedir (Şekil 3.1). ARM64'te kullanılan komut setlerinden bazıları şunlardır:

1. A64: Bu, ARM64'te kullanılan ana komut seti olup, 64 bit işlemi desteklemektedir. Mantıksal, aritmetik ve kontrol işlemleri için geniş bir komut yelpazesi içermektedir. A64, işlemcinin verimli bir şekilde çalışmasını sağlamak amacıyla optimize edilmiş olup, enerji verimliliği, güvenlik ve yüksek performans gibi tasarım ilkelerine odaklanmaktadır. Buna ek olarak, A64, ARM64 mimarisinde yazılmış uygulamaların yürütülmesi için gereklidir ve ARM tabanlı cihazlarda yaygın olarak kullanılmaktadır.

2. A32/T32: Önceden ARMv7 olarak bilinmekte olan bu mimariler ARM mimarisinin 32 bitlik komut setleridir ve 32 bitlik işlemcilerde kullanılmaktadırlar. Mantıksal, aritmetik ve kontrol işlemleri için komutlar içermektedir. T32, A32'nin alt kümesi olup, daha az bellek alanı kullanarak daha az karmaşık işlemleri gerçekleştirmek amacıyla tasarlanmıştır. Her ne kadar

A32/T32, A64 komut seti kadar güçlü olmadığı bilinse de hala birçok cihazda yaygın olarak kullanılmaktadır. Ayrıca, bazı ARM64 işlemcileri de A32/T32 desteği sağlamaktadır. A32/T32, özellikle daha düşük maliyetli veya daha düşük performans gereksinimleri olan cihazlarda tercih edilen bir seçenektir.

3. SIMD: SIMD, Tek Komutla Çoklu Veri komut seti olarak da bilinmekte ve özellikle multimedya ve sinyal işleme gibi belirli hesaplama türlerini hızlandırmak için kullanılmaktadır. SIMD komutları, verilerin paralel işlenmesini destekleyen komutlar olarak öne çıkmaktadır. Bu sayede, aynı anda birçok veri öğesi üzerinde işlem yapılabilmekte ve hesaplama süresi kısaltılabilmektedir.

Özellikle multimedya ve sinyal işleme gibi alanlarda, çok sayıda verinin işlenmesi gerektiği için SIMD komutlarından oldukça faydalanılmaktadır. Örneğin, bir resmin piksel değerlerini işlemek veya bir ses dosyasındaki örnekleri işlemek için SIMD komutları kullanılabilmektedir. Bu işlemler, her piksel veya her örnek için ayrı ayrı hesaplamalar yapmak yerine, tüm pikseller veya örnekler üzerinde aynı işlemi uygulayarak çok daha hızlı bir şekilde gerçekleştirilebilmektedir.

Özellikle yüksek performans gerektiren uygulamalar için önemli olan SIMD komutları, çoğu modern işlemci tarafından desteklenmekte ve özellikle grafik işlemcileri ve veri merkezi sunucuları gibi yüksek performanslı sistemlerde yaygın olarak kullanılmaktadır.

4. Kriptografi: Kriptografi komut seti, şifreleme ve deşifreleme gibi kriptografik işlemleri gerçekleştirmek için komutlar içermektedir. Bu komut seti, güvenli iletişim ve dijital imza gibi uygulamalar için hızlı ve güvenli bir şekilde işlem yapabilmek için özel olarak optimize edilmiş olup, güvenli iletişim ve dijital imzalar gibi uygulamalarda kullanılmaktadır. Kriptografi komutları, örneğin AES (Advanced Encryption Standard) veya DES (Data Encryption Standard) gibi şifreleme algoritmalarını hızlandırmak için

kullanılabilmektedir. Bu sebeplerde Kriptografi komut seti, özellikle güvenli iletişim ve veri koruma gibi konularda büyük önem taşımaktadır.

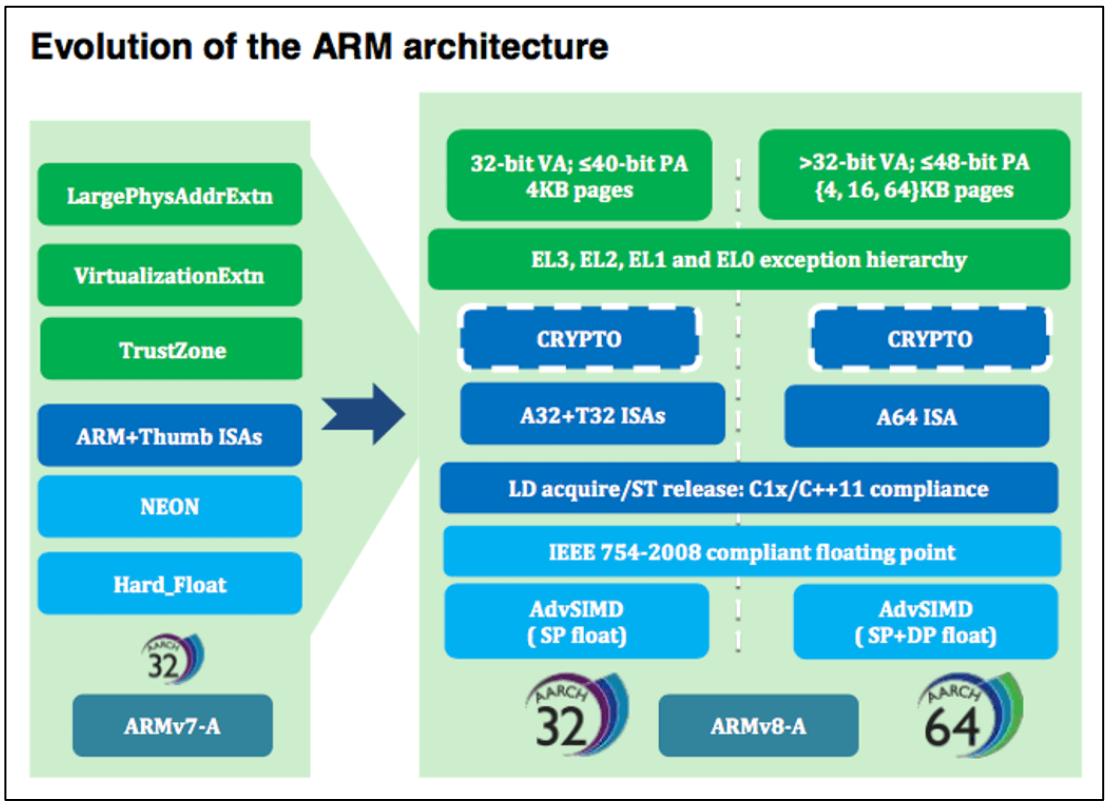
5. Sanallaştırma: Sanallaştırma komut seti, donanım kaynaklarının sanallaştırılması için kullanılan komutları içermektedir. Bu komutlar, tek bir işlemci üzerinde birden fazla işletim sistemi veya sanal makinenin çalıştırılmasına izin vermekte ve sistem kullanımını artırarak maliyetleri düşürmeye yardımcı olmaktadırlar.

Sanallaştırma komut seti, öncelikle sunucu sistemlerinde; özellikle sanal özel sunucu (VPS) hizmetlerinde yaygın olarak kullanılmaktadırlar. Sanallaştırma teknolojisi, donanımın daha etkili kullanılmasına ve iş yüklerinin daha iyi yönetilmesine yardımcı olmasının yanı sıra, birden fazla işletim sistemi veya uygulama aynı anda çalıştırılabilmesi sayesinde daha esnek bir işlem ortamı sağlamaktadır.

6. TrustZone: ARM TrustZone teknolojisi, mobil ve gömülü sistemler için donanım tabanlı güvenlik sağlaması için tasarlanmış olup, bu teknolojiyi desteklemeyi amaçlayan komutlar içermektedir. TrustZone, hassas verilerin depolanması için güvenli alanlar oluşturmak ve sistem kaynaklarına izinsiz erişimi önlemek için kullanılabilmektedir. Oluşturulan bu alanlar, sistem kaynaklarına izinsiz erişimi engelleyerek veri güvenliğini sağlamaktadır. TrustZone teknolojisi, cihazların bölünmüş bir yapıda çalışmasına yardımcı olmakta, böylece bir bölüm güvenli bir şekilde çalışırken diğer bölüm güvenli olmayan işlemler için kullanılabilmektedir.

Genel olarak bakıldığında; ARM64 ISA (Industry Standard Architecture), çeşitli hesaplama türlerini ve sistem gereksinimlerini karşılamak için tasarlanmış bir dizi komut setini içermektedir. Bu komut setleri, birleştirilerek ve özelleştirilerek, belirli uygulamalar veya kullanım durumları için optimize edilmiş işlemciler oluşturulabilir. A64 komut seti, ARM64 işlemcilerinde kullanılan ana komut setidir ve 64 bit işlemi desteklemektedir. Ayrıca, A32/T32 komut setleri, ARM32 ve bazı ARM64 işlemcilerinde desteklenen 32 bitlik komut setleridir. SIMD komut seti, verilerin paralel

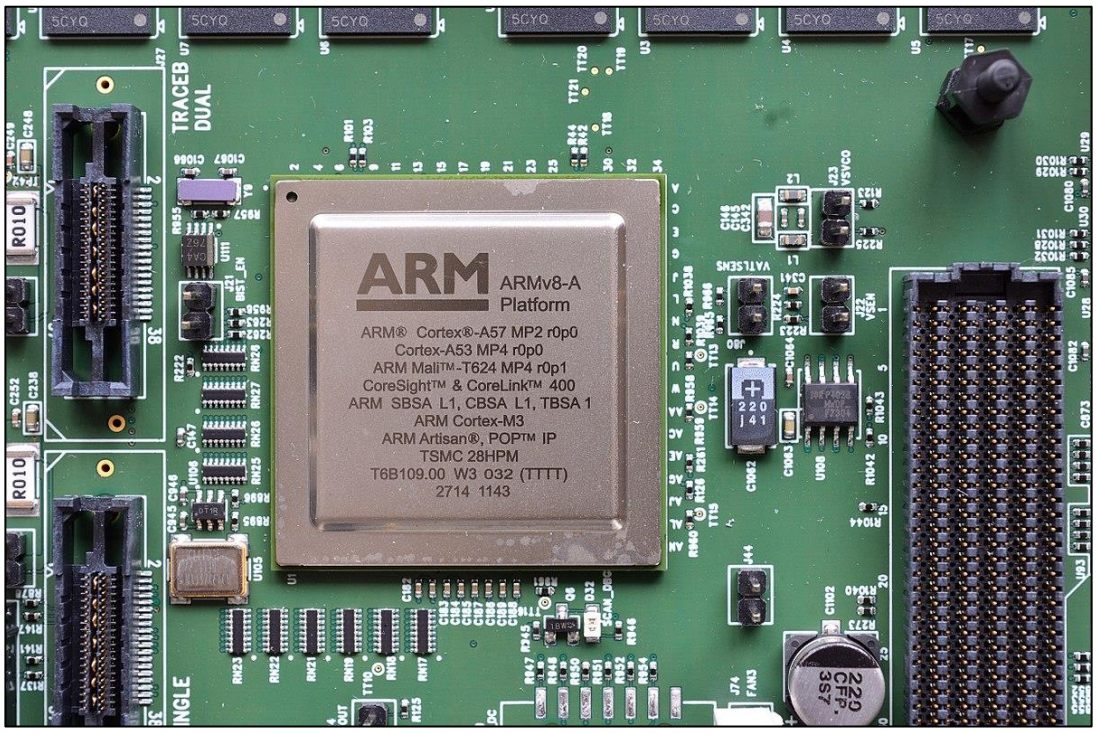
işlenmesini destekleyen Tek Komutla Çoklu Veri (SIMD) komutlarını içerir ve multimedya ve sinyal işleme gibi belirli hesaplama türlerini hızlandırmak için kullanılabilmektedir. Kriptografi komut seti, şifreleme ve deşifreleme gibi kriptografik işlemleri gerçekleştirmek için tasarlanmıştır ve güvenli iletişim ve dijital imzalar gibi uygulamalarda kullanılabilmektedir. Sanallaştırma komut seti, donanım kaynaklarının sanallaştırılmasını desteklemek için komutlar içermekte ve tek bir işlemci üzerinde birden fazla işletim sistemi veya sanal makinenin çalıştırılmasına izin vermektedir. Son olarak, TrustZone komut seti ise, mobil ve gömülü sistemler için donanım tabanlı güvenlik sağlayan ARM TrustZone teknolojisini desteklemek için komutlar içermektedir.



Şekil 3.1. ARM Mimarisinin Dönüşümü

ARMv8-A mimarisinin (Şekil 3.2), önceki ARMv7-A mimarisine kıyasla birçok yenilik getirdiği bilinmektedir. Bu yeniliklerin en önemlisi 64-bit işlemci desteğidir. Bu sayede daha büyük bellekler kullanılabilmekte ve daha fazla veri işlenebilmektedir. Bunlara ek olarak, ARMv8-A mimarisi daha gelişmiş bellek yönetimi mekanizmaları,

daha iyi işlem performansı ve daha güçlü güvenlik özellikleri sunmaktadır. Bu özellikler sayesinde ARMv8-A mimarisi, birçok farklı uygulama alanında kullanılmakta, hassas verilerin daha iyi korunması ve izinsiz erişimlere karşı daha güçlü bir koruma sağlanmaktadır. Özellikle mobil cihazlar, sunucular, yapay zekâ, makine öğrenmesi ve otonom araçlar gibi alanlarda sıklıkla tercih edilmektedir. ARMv8-A mimarisine geçiş, daha yüksek performans, daha iyi güvenlik ve daha ölçeklenebilir bir işlemci mimarisi sağlamaktadır ve bu, işlemcilerin daha fazla sayıda iş parçacığını eşzamanlı olarak işleyebilmesi sayesinde mümkün olmaktadır. Sonuç olarak, ARM tabanlı sistemler için ARMv8-A mimarisine geçiş, önemli bir gelişme olarak nitelendirilmektedir.

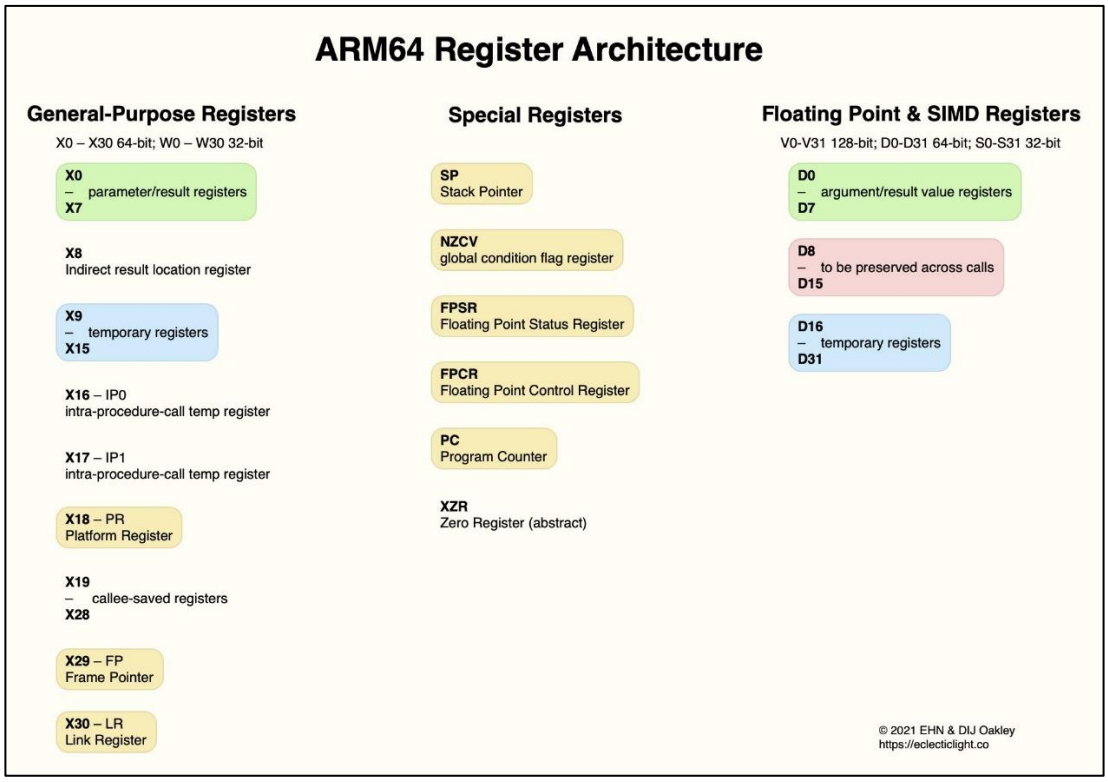


Şekil 3.2. Cortex-A57/A53 ile Armv8-A Platformu [16]

3.1. ARM64 Tabanlı Mikroişlemcilerde Registerların Rolü

ARM64 mimarisi (Şekil 3.3), mikroişlemcilerin performansı ve verimliliği için önemli bir faktör olan register dosyalarına büyük önem vermektedir. Registerlar, mikroişlemcinin sıklıkla eriştiği verileri depolamak için kullanılan küçük bellek

alanları olarak nitelendirilmektedir. ARM64 mimarisi, register dosyalarının gücü verimli kullanması ve gecikme oranının düşük olması amacıyla özel olarak tasarlanmıştır. Ayrıca, büyük register dosyaları için register yeniden adlandırma, tahmin, spekülasyon ve yazılım boru hattı desteği gibi özellikler sunmaktadır. Bu özellikler sayesinde, mikroişlemcinin performansı ve verimliliği önemli ölçüde artırılarak işlemci iş yükü daha verimli bir şekilde yönetilebilmektedir.



Şekil 3.3. ARM64 Register Mimarisi [15]

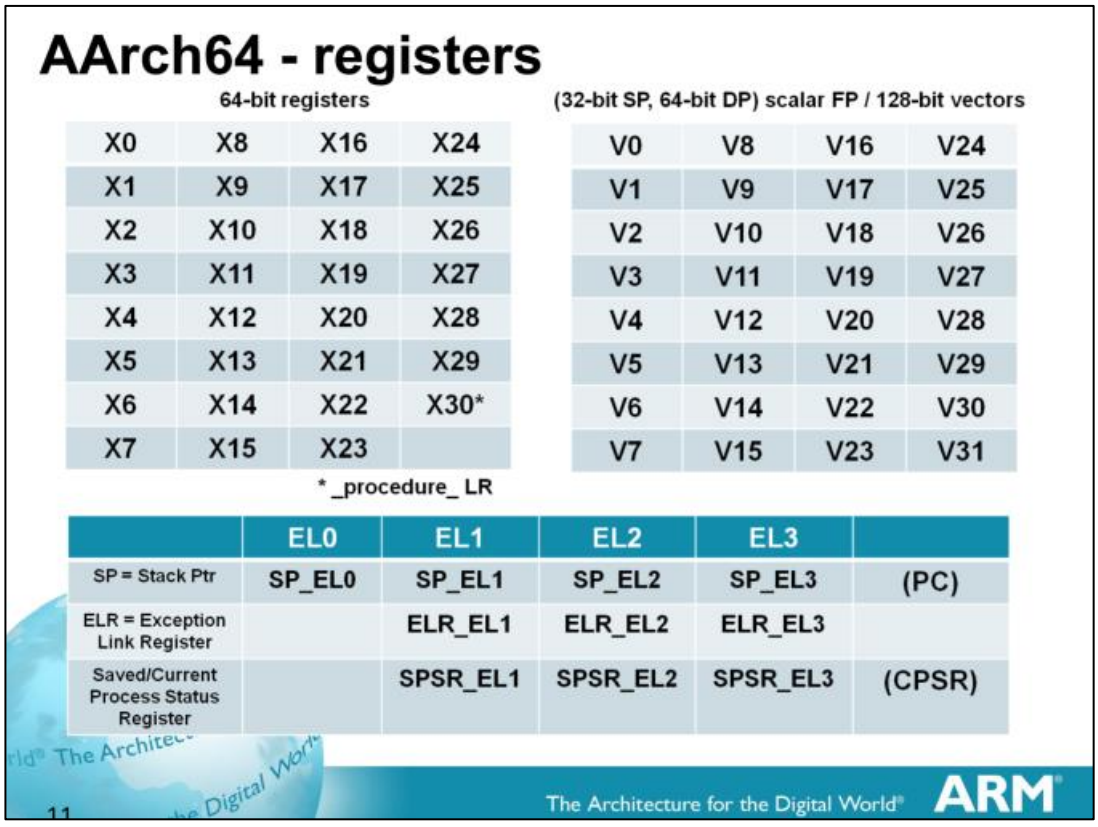
3.2. ARM64 Mimarisinde Yer Alan Register Sayı ve Boyutları

3.2.1. Genel amaçlı registerlar

ARM64 tabanlı mikroişlemcilerin (Şekil 3.4) en önemli özelliklerinden biri, 31 adet 64 bit genişliğinde genel amaçlı yazmaçtan oluşan register dosyasına sahip olmasıdır. Bu yazmaçlar, X0'dan X30'a kadar sıralı olarak adlandırılmakta olup; aritmetik, mantıksal ve diğer işlemler için veri depolamak için kullanılmaktadır. X31 yazmacı, yığın işaretçisi için ayrılmıştır. Her bir registerın özel bir amaç için optimize edilmiş

olması, ARM64 mimarisinin yüksek performans ve verimlilik sağlamasına yardımcı olmaktadır. Örneğin, X0 kaydı bir işlevin ilk argümanını tutmak için kullanılırken, X19 ve X20 yazmaçları uzun çiftlik aritmetiği işlemleri için optimize edilmiş olabilmektedir.

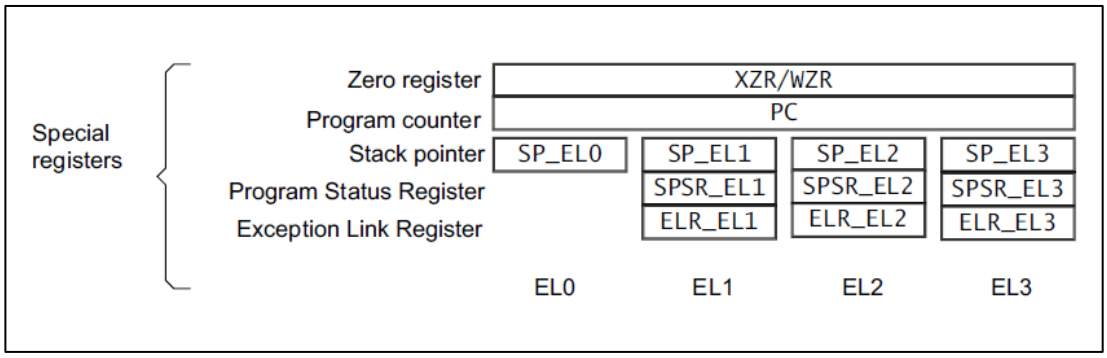
ARM64 mimarisi, register dosyasının boyutunu küçük tutarak register erişim hızını artırmayı hedeflemektedir. Bu sayede, sık kullanılan verilere daha çabuk erişim sağlanmakta ve mikroişlemcinin performansı artırılmaktadır. Bu özellikler, yüksek performanslı bilgisayarlar ve cihazlar tasarlamak için kritik öneme sahiptir. Sonuç olarak, ARM64 tabanlı mikroişlemcilerde register dosyasının rolü, sıklıkla erişilen verilerin hızlı erişimine olanak tanıyarak yüksek performans ve verimlilik sağlamaktır.



Şekil 3.4. ARMv8 A64 ISA Registerları [23]

3.2.2. Özel registerlar

ARM64 tabanlı mikroişlemcilerde, genel amaçlı registerlara ek olarak program sayacı (PC), durum kodu kaydı (CCR) ve istisna bağlantı kaydı (ELR) gibi birkaç özel amaçlı register (Şekil 3.5) bulunmaktadır. CCR, aritmetik ve mantıksal işlemlerin sonuçlarını gösteren durum bayraklarını saklamak için kullanılırken, PC yürütülen komutun adresini depolamak için kullanılmaktadır. ELR, istisnalar için dönüş adresini saklamak için kullanılmaktadır. Registerların işlevleri arasında, mantıksal ve aritmetik işlemler için verileri tutmak, bellek işlemleri için adresleri depolamak, işlev argümanlarını ve dönüş değerlerini saklamak yer almaktadır.



Şekil 3.5. Aarch64 Özel Registerlar

İşaretçi kaydı (SP) da 64-bit büyüklüğünde olup, yığın göstericisi olarak kullanılmaktadır. SP'ye çoğu talimat tarafından başvurulamadığı gibi, bazı aritmetik komut biçimleri; örneğin ADD komutu, bir işlevdeki yığın işaretçisini ayarlamak için geçerli yığın işaretçisini okuyabilmekte ve yazabilmektedir. Status register (PSTATE) kaydı, işlem durumu bilgilerini içeren ve 32-bit büyüklüğünde bir registerdır. Tablo 3.1 registerların boyutlarıyla illgili bilgileri içermektedir.

Zero register, kaynak register olarak kullanıldığında sıfır olarak okumakta ve hedef kayıt olarak kullanıldığında sonucu atmaktadır. Zero register ise çoğu talimatta kullanılsa da tümünde kullanılamamaktadır.

Name	Size	Description
WZR	32 bits	Zero register
XZR	64 bits	Zero register
WSP	32 bits	Current stack pointer
SP	64 bits	Current stack pointer
PC	64 bits	Program counter

Tablo 3.1. Aarch64'te Özel Registerlar

Register sayısı, yüksek performanslı bilgisayarlar tasarlamak için kritik öneme sahiptir; bu nedenle register erişiminin daha hızlı olabilmesi için register dosyasının boyutunun küçük olması gerekmektedir. Dolayısıyla, ARM64 tabanlı mikroişlemcilerde kaydedicilerin rolü, bu mikroişlemcilerin yüksek performans ve verimliliğine katkıda bulunan, sık kullanılan verilere hızlı erişim sağlamaktır.

Registerlar, işlemcinin hızı ve performansı için önemli bir faktördür ve işlemcinin hafıza yönetimi ve veri işleme yeteneklerinin de büyük ölçüde iyileştirilmesine yardımcı olmaktadırlar. Bu özel amaçlı registerlar, özellikle de program sayacı ve durum kodu kaydı, işlemcinin işletim sistemleri ve uygulama yazılımları gibi temel bileşenlerinin doğru bir şekilde çalışmasını sağlamaktadır [13].

3.2.3. SIMD registerlar

"SIMD" ise, Single Instruction Multiple Data'nın kısaltmasıdır ve birden fazla veri öğesinde aynı işlemi yapabilen bir işlem birimi türünü ifade etmektedir. SIMD registerlar, SIMD işlemlerini gerçekleştirmek için kullanılan özel registerlardır. Bu registerlar, işlemcideki genel amaçlı registerlar gibi çalışmaktadır, ancak SIMD işlemleri için optimize edilmiştir.

SIMD işlemleri, birçok veri öğesi üzerinde aynı işlemi aynı anda yaparak, işlemci performansını arttırmaktadır.

3.3. ARM64 Register Fonksiyonları

ARM64 tabanlı mikroişlemcilerdeki registerlar, öncelikli olarak aritmetik ve mantıksal işlemler için veri depolama görevi görmektedir. Bu registerlar, çeşitli veri tiplerini saklamak için kullanılabilmektedir; örneğin, tam sayılar, kayan nokta sayıları, adresler vb. İşlemci, verileri registerlardan okuyarak, işlemler yapmakta ve sonuçları yine registerlara yazmaktadırlar.

Bunun yanı sıra, registerlar bellek işlemleri için adresleri tutma işlevini de görmektedirler. Bellek işlemleri, bellekteki verilerin nerede saklandığını gösteren bellek adreslerini gerektirdiğinden; registerlar, bellek adreslerini tutmak için kullanılmakta ve işlemci, bellekteki verilere erişmek için bu adresleri kullanmaktadır.

Registerlar ayrıca, işlev argümanlarını ve dönüş değerlerini tutmak için kullanılmaktadır. Bir işlev çağrısı sırasında, argümanlar genellikle registerlarda saklanmakta ve işlemci, argümanları registerlardan okuyarak işlevin çalışması için kullanmaktadır. İşlev geri döndüğünde, dönüş değeri genellikle bir registera yazılmakta ve işlemci, dönüş değerini bu registertan okumaktadır.

Registerlar, hızlı erişim sağlamalarıyla da dikkat çekmektedirler. İşlemcinin hızlı bir şekilde erişebileceği küçük bir bellek alanı sağlamaları sebebiyle, işlemcinin daha hızlı çalışmasına ve programların daha hızlı yürütülmesine yardımcı olmaktadırlar. Tüm bu nedenlerden ötürü, registerlar işlemcinin hızı ve performansı için kritik bir faktör olduğu bilinmektedir. Register dosyasının boyutunun küçük tutulması, register erişiminin hızlı olabilmesi için önemlidir. ARM64 tabanlı mikroişlemcilerdeki registerların önemli rolü, bu mikroişlemcilerin yüksek performans ve verimliliğine katkıda bulunarak sık kullanılan verilere hızlı erişim sağlamaktır [17] [18].

3.4. ARM64 Kayıtlarının Kaydetme, Yükleme, İşleme ve Transfer Fonksiyonları

ARM64 mimarisinin, genel amaçlı kayıtların çeşitli fonksiyonları olduğu 64-bit işlemci olduğu bilinmektedir. Bu kayıtlar, işlemcinin performansını artırmak ve

işlemci işlevselliğini desteklemek için tasarlanmıştır. Genel amaçlı kayıtların temel işlevleri, verileri ve adresleri tutmak, verileri işlemek ve kaydetmek, işaretçileri ve işlem durumu bilgilerini saklamak olarak sıralanabilir. Genel amaçlı kayıtlar, programlama dillerinde değişkenlerin değerlerini saklamak için kullanılmakta ve işlemci, bu kayıtlarda depolanan verilere erişerek işlemler yapmaktadır.

Genel olarak, ARM64 mimarisindeki genel amaçlı kayıtlar, işlemcinin veri işleme ve yönetme yeteneklerini desteklemekte ve işlemcinin performansını artırmaya yardımcı olmaktadır. ARM64 kayıtlarının temel fonksiyonlarına (Şekil 3.6) ilişkin özet bilgiler aşağıda yer almaktadır:

Kaydetme (store) işlemleri: Genel amaçlı kayıtlardaki değerleri bellekteki bir yere kaydetmek amacıyla kullanılmaktadır. Bu işlem için genellikle "STR" (store register) veya "STUR" (store register with unscaled offset) komutları kullanılmaktadır.

- MOV: Kaydedilecek veriyi bir kayıttan diğerine taşımak için kullanılmaktadır.
- STR: Bir kayıttaki veriyi belleğe kaydetmek için kullanılmaktadır.
- STP/LDP: Birden fazla kayda aynı anda erişmek için kullanılmaktadır.

Yükleme (load) işlemleri: Bellekteki bir yeri genel amaçlı kayıtlara yüklemek için kullanılmaktadır. Bu işlem için genellikle "LDR" (load register) veya "LDUR" (load register with unscaled offset) komutları kullanılmaktadır.

- LDR: Bellekten veri okumak ve bir kayda yüklemek için kullanılmaktadır.
- LDRB/LDRH/LDRSB/LDRSH: Bellekten okunan veriyi işlemeye uygun formatta bir kayda yüklemek için kullanılmaktadır.
- LDRSW: 32-bit imzalı tamsayı değerlerini bir kayda yüklemek için kullanılmaktadır.

İşleme (arithmetic and logical operations) işlemleri: Genel amaçlı kayıtlar arasında çeşitli aritmetik ve mantıksal işlemler gerçekleştirilebilmektedir. Bu işlemler için toplama, çıkarma, çarpma, bölme, and, or, xor gibi komutlar kullanılmaktadır.

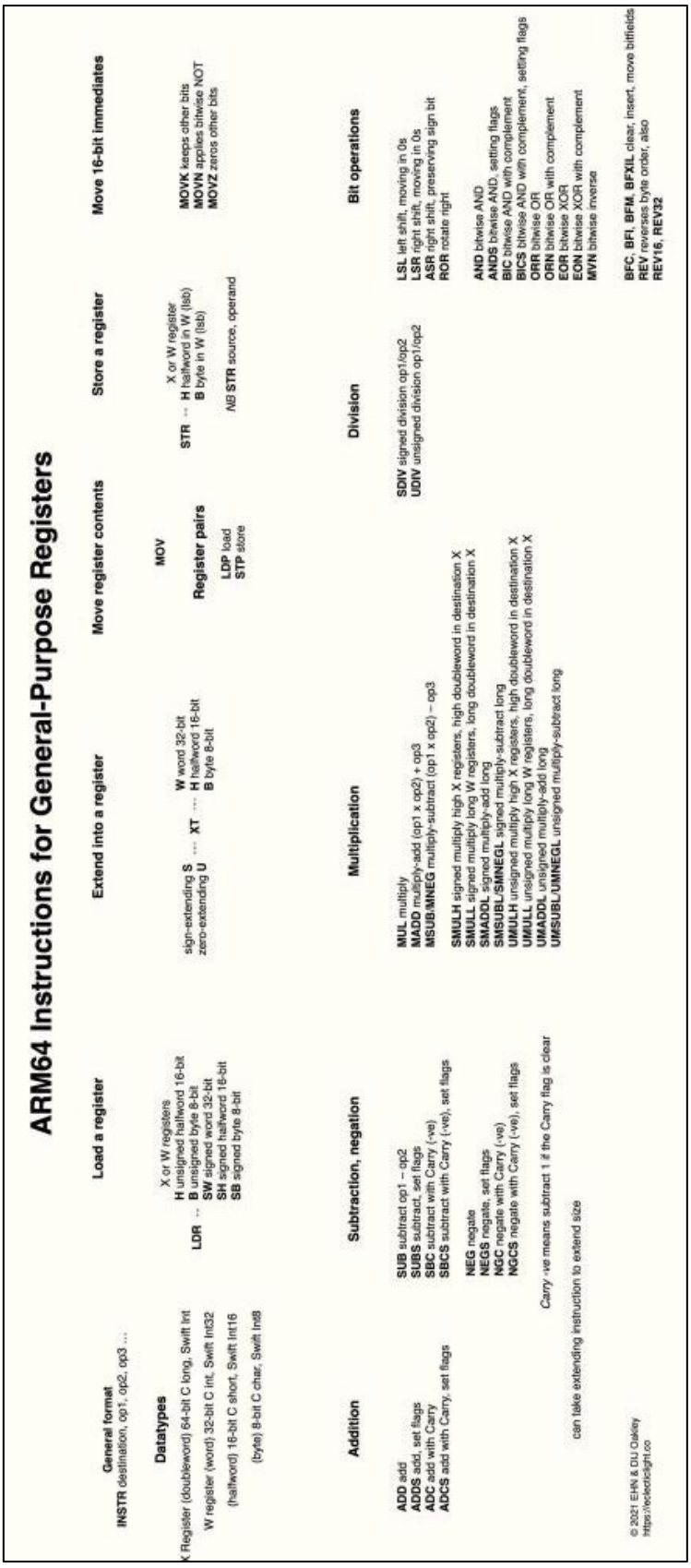
- ADD/SUB: İki sayıyı toplama ya da çıkarma amacıyla kullanılmaktadır.
- MUL/SDIV/UDIV: Çarpma ve tam sayı bölme işlemleri amacıyla kullanılmaktadır.
- AND/OR/XOR: Mantıksal AND/OR/XOR işlemleri amacıyla kullanılmaktadır.
- ASR/LSL/LSR: Bit kaydırma işlemleri için kullanılmaktadır.
- CMP: İki sayının karşılaştırması için kullanılmaktadır.

Transfer (move) işlemleri: Genel amaçlı kayıtlar arasında değerlerin kopyalanması için kullanılmaktadır. Bu işlem için genellikle "MOV" (move) ya da "ORR" (bitwise OR) komutları kullanılmaktadır.

- BR/B: Program akışını kontrol etmek amacıyla kullanılmaktadır.
- BL/BLR: Alt programlara çağrı yapmak amacıyla kullanılmaktadır.
- RET: Bir alt programdan dönüş yapmak amacıyla kullanılmaktadır.
- CBZ/CBNZ: Bir kaydın sıfır veya sıfır olmayan bir değere eşit olup olmadığını kontrol etmek amacıyla kullanılmaktadır.

Bu işlevler, geniş kategoriler altında da gruplandırılabilmektedir; kaydetme, yükleme, işleme ve transfer. Kaydetme işlevi, verileri bir kayıt içine yazmak için kullanılmaktadır. Yükleme işlevi, bellekten verileri okumak ve bir kayda yüklemek için kullanılmaktadır. İşleme işlevi, kayıtlar arasında aritmetik veya mantıksal işlemler gerçekleştirmek için kullanılmaktadır. Transfer işlevi ise, kayıtlar arasında veri taşımak için kullanılmaktadır. Ancak, bu sadece genel bir kategorilendirme şekli olup, ARM64 mimarisindeki farklı komutlar, bu işlemleri farklı şekillerde gerçekleştirebilmektedir. Örneğin, bir ADD komutu, iki kayıt arasında bir toplama işlemi gerçekleştirirken, bir LDR komutu, bellekten bir veriyi yükleyip bir kayda

yerleştirmek için kullanılabilmektedir. Çok sayıda genel amaçlı kayıt içen ve her biri, işlemcinin hızını ve performansını artırmak için önemli bir rol oynayan bu kayıtlar, hızlı şekilde erişilebilen küçük bellek alanlarıdır ve işlemciye veri ve adresleri hızlı bir şekilde işlemesi için olanak sağlamaktadır. Bu sebeplerle, ARM64 mimarisi, yüksek performanslı mikroişlemciler için önemli bir seçenek olarak kabul edilmektedir.



Şekil 3.6. Genel Amaçlı Kayıtlar İçin Arm64 Talimatları [22]

3.5. ARM64 Kayıtlarının Öncelik Sırası, Erişim Modları ve Kısıtlamaları

ARM64, 31 adet 64-bit genel amaçlı kayıtlar içermektedir. Bu kayıtlar, öncelik sırasına göre X0-X7, X8-X15, X16-X23, X24-X30 ve SP (Stack Pointer) olarak gruplandırılmaktadır. Kayıtların erişim modları, kullanılan işlem türüne bağlı olarak değişebilmektedir. Örneğin, bazı işlemler yalnızca kayıtların alt kısımlarına erişebilirken, diğerleri tam kayıtlara erişebilmektedir [21].

Öncelik sırası: Kayıtlar, işlem yapılacak verinin türüne ve boyutuna göre seçilmektedirler. Örneğin, küçük 8-bit verileri saklamak için X0-X7 kayıtları tercih edilirken, daha büyük 64-bit verileri saklamak için X0-X30 kayıtları kullanılabilir. Ayrıca, bazı kayıtlar özel amaçlı olup, belirli görevleri yerine getirmek için tasarlanmıştır. Örneğin, program sayacı (PC) kaydı, işlemcinin programın hangi bölümünde olduğunu takip etmek için kullanılmaktadır. Diğer özel kayıtlar arasında SP (Stack Pointer), XZR (Zero Register), FP (Frame Pointer), LR (Link Register) gibi kayıtlar bulunmakta, bu özel kayıtlar, işlem yapmak için özel komutlar aracılığıyla kullanılmaktadır.

Erişim modları: Kayıtların erişim modları, genellikle kullanıldıkları işleve ve kayıt türüne bağlı olarak değişmektedir. Birkaç farklı erişim modu bulunmaktadır. Bazı kayıtlar salt okunur (read-only) olup, yalnızca belirli bir işlem için gerekli verileri sağlamak için kullanılmaktadır. Bazıları sadece yazılabilir (write-only) olup, genellikle işlem sonuçlarını kaydetmek için kullanılmaktadır. Bazıları ise hem okunabilir hem de yazılabilir (read-write) ve işlemde kullanılan verileri saklamak ve sonuçları kaydetmek için kullanılmaktadır. Tüm bunlara ek olarak, bazı kayıtlar ise tamamen özel amaçlı ve yalnızca belirli işlemler için kullanılan kayıtlardır. Bu özel kayıtlar, genellikle işaretçileri veya işlem durumu bilgilerini tutmak için tasarlanmıştır.

Kısıtlamalar: Kayıtların kullanımı belirli kısıtlamalara tabi olabilmektedir. Örnek vermek gerekirse, özel amaçlı kayıtlar, belirli görevler için tasarlanmıştır ve genel amaçlı işlemlerde kullanılamamaktadırlar. Buna ek olarak, kimi kayıtların erişim

modları, işlem yapılan verinin türüne ve boyutuna bağlı olarak değişebilmektedir. Bu duruma örnek olarak 8-bit bir veri için X0-X7 kayıtları tercih edilmesine rağmen 64-bit bir veri için X0-X30 kayıtları kullanılması verilebilmektedir. Kayıtların erişim modları ve kullanımı, işlemcinin çalıştığı mod (Kernel modu ya da kullanıcı modu) ve güvenlik gereksinimlerine göre değişebilmektedir [21].

3.6. ARM64 Kayıtlarının Performans ve Enerji Yönetimi Üzerindeki Etkisi

ARM64 işlemcileri, veri saklama, işleme ve transfer etme işlemleri için kayıtları kullanmaktadırlar. Kayıtlar, performans ve enerji yönetimi açısından önemli bir rol oynamakla birlikte kayıtların boyutu, sayısı ve kullanımı da önem arz etmektedir. Çünkü, performans açısından, kayıtların boyutu, veri erişim hızını etkileyebilmektedir. ARM64 işlemcileri, kayıtlardaki verilere çok hızlı bir şekilde erişebildikleri için kayıtların boyutu arttıkça, işlemcinin veri erişim hızı da artmakta ve böylece işlemcinin performansı artmaktadır. Örneğin, 64-bit verileri saklamak için 31 genel amaçlı kayıt bulunmakta olup, bunlar X0-X30 olarak adlandırılmaktadır. Bu kayıtlar, veri erişim hızını artırarak işlemcinin performansını artırmaktadır.

Enerji yönetimi açısından, kayıtların kullanımı, işlemcinin enerji tüketimini etkileyebilmektedir. Kayıtların kullanımı, işlemcinin bellek erişimine olan ihtiyacını azaltarak işlemcinin enerji tüketimini düşürmektedir. Fakat, kayıtların boyutu arttıkça, işlemcinin enerji tüketimi de arttığından, kayıtların boyutu ve sayısının, enerji verimliliği açısından dikkatli bir şekilde tasarlanması gerekmektedir. Ek olarak, ARM64 işlemcilerinde kayıtların kullanımı, yazılımın optimize edilmesi konusunda etkileyici özelliklere sahiptir. İyi optimize edilmiş bir yazılım, kayıtları etkin bir şekilde kullanarak işlemcinin performansını arttırabilmekte ve böylece enerji tüketimini düşürebilmektedir. Bu nedenle, yazılım geliştiricilerinin, kayıtların kullanımını ve yönetimini dikkate alarak optimize edilmiş yazılımlar yazması önem arz etmektedir.

Sonuç olarak, ARM64 kayıtları, işlemcinin performansı ve enerji tüketimi üzerinde önemli bir etkiye sahip olduğu söylenebilir. Kayıtların boyutu, sayısı ve kullanımı,

işlemcinin performansını, enerji tüketimini ve optimize edilmiş yazılımın performansını etkileyebilir. Bu nedenle, kayıtların dikkatli bir şekilde tasarlanması ve kullanılması, işlemcinin performansı ve enerji verimliliği açısından önemlidir.

3.7. ARM64 Register Özellikleri

ARM64 tabanlı mikroişlemcilerdeki registerların genişlik, isimlendirme, erişim hızı, kaynak yönetimi, güç verimliliği, register sıralaması, register durumu registerları, kayıt adresleme, kayıt önceliği, kayıt performansı, vektör kaydedicileri, gölge kaydedicileri, özel amaçlı kaydediciler ve yığın kaydedicileri gibi özellikleri bulunmaktadır. Bunlar aşağıdaki gibi açıklanmaktadır:

1. *Genişlik:* ARM64 tabanlı mikroişlemcilerdeki her bir register, 64 bit genişliğinde olduğundan, önceki 32 bitlik registerlara göre iki kat daha fazla veri saklayabilmektedir.
2. *İsimlendirme:* ARM64 tabanlı mikroişlemcilerdeki registerlar, X0 ile X30 arasında etiketlenmekte ve böylece her bir kaydın benzersiz bir numaralandırma sistemine sahip olması yerine daha anlamlı bir isimlendirme sistemine sahip olmaktadırlar.
3. *Erişim Hızı:* Registerlar, işlemcinin daha çabuk şekilde erişebileceği bir bellek bloğudur ve bu sebeple diğer bellek bölgelerine göre daha hızlı bir şekilde erişilebilmektedirler.
4. *Kaynak Yönetimi:* ARM64 tabanlı mikroişlemciler, sınırlı sayıda registera sahip olduğundan, kaynak yönetimi önem arz etmektedir. Bu yönetim için register havuzları ve register renklendirmesi gibi yöntemler kullanılabilmektedir.
5. *Güç Verimliliği:* Güç Verimliliği: Register kullanımı, diğer bellek bölgelerine göre daha az güç tüketilmesini sağladığından, ARM64 tabanlı mikroişlemciler daha enerji verimli olmaktadırlar.
6. *Register Sıralaması:* ARM64 tabanlı mikroişlemciler, registerları belirli amaçlar için ayrılmış register setleri olarak düzenlenmektedir. Bu sayede, her register seti belirli bir görevi yerine getirmek için optimize edilir. Örneğin, X0-X7 kaydedicileri genellikle işlev argümanları ve dönüş

değerleri için ayrılırken, X8-X15 kaydedicileri çizim işlemleri için geçici kaydediciler olarak kullanılmaktadır. X16-X30 kaydedicileri ise işlev çağrıları sırasında kullanılmakta olup, işlemcinin çalışma zamanındaki durumunu izlemek için durum kaydedicileri kullanılmaktadır.

7. *Durum Registerları:* ARM64 tabanlı mikroişlemcilerde, kaydedicilerin bazılarının işlem sırasında kullanılıp kullanılmadığını belirlemek için durum kaydedicileri kullanılmaktadır. Bu durum kaydedicileri, yığın alanları ve kaydedicilerin atama geçmişini tutmaktadır.
8. *Kayıt Adresleme:* ARM64 kayıtları, bellek adreslemesi için kullanılabilecek birçok farklı yönteme sahiptir. Bu adresleme yöntemleri arasında öteleme (offset), indisli (indexed), ölçekli (scaled) ve indisli-ölçekli (indexed-scaled) adresleme bulunmaktadır.
9. *Kayıt Önceliği:* ARM64 kayıtları, belirli bir öncelik sırasına göre erişilebilmektedir. Örneğin, bazı kayıtlar daha hızlı erişilebilirken, diğerleri daha yavaş erişilebilirdir.
10. *Kayıt Performansı:* ARM64 kayıtları, performans ve güç tüketimi açısından önemli bir rol oynamaktadır. Kayıtların etkin kullanımı, işlemcinin performansını arttırabilmekte ve güç tüketimini azaltabilmektedir. Kayıtların kullanımının yanı sıra, işlemcinin bellek erişimi, önbellek kullanımı ve diğer faktörler de performansı etkileyebilmektedir.
11. *Vektör Kaydedicileri:* ARM64 tabanlı mikroişlemcilerdeki vektör kaydedicileri, aynı türdeki birden çok veri öğesini aynı anda işlemek için kullanılmaktadır. Bu özellik, SIMD (Single Instruction Multiple Data) işlemlerinin hızlı bir şekilde gerçekleştirilmesine olanak tanımaktadır.
12. *Gölge Kaydedicileri:* ARM64 tabanlı mikroişlemcilerdeki gölge kaydedicileri, işlem sırasında kullanılan kaydedicilerin bir kopyasını tutmaktadır. Bu, bir işlev çağrısının ardından kaydedicilerin değiştirilmesi durumunda, önceki değerlerin geri yüklenmesini kolaylaştırmaktadır.
13. *Özel Amaçlı Kaydediciler:* ARM64 tabanlı mikroişlemcilerde, bazı kaydediciler özel amaçlar için ayrılmıştır.

14. *Yığın Kaydedicileri:* ARM64 tabanlı mikroişlemcilerdeki yığın kaydedicileri, işlem sırasında geçici verileri saklamak için kullanılmaktadır. Yığın kaydedicileri, özellikle işlem sırasında değişen verileri takip etmek için kullanışlı olarak nitelendirilmektedir.

Bu özellikler sayesinde, ARM64 tabanlı mikroişlemcilerdeki registerların çok yönlü ve esnek bir şekilde kullanılması olanaklı hale gelmektedir. Kaydediciler, hızlı ve güçlü işlem yapması amacıyla gereken verileri depolamak için kullanılmakta ve birçok işlem için gerekli olan kaynakları yönetmek için çeşitli teknikler kullanılmaktadır [19].

3.8. ARM64 Registerları, Kullanım Alanları ve Özellikleri

Genel Amaçlı Registerlar (X0-X30): Bu registerlar, 64-bitlik genel amaçlı verileri depolamak için kullanılmaktadır. Programlama dilindeki değişkenlerin depolanması ve işlenmesi için kullanılmaktadır. Fonksiyon çağrıları sırasında parametreler bu registerlara yüklenmekte ve registerlardan değerler geri döndürülmektedir.

Program Sayacı Registerı (PC): Bu register, işlemcinin mevcut komut satırının adresini tutmaktadır. Bir program çalıştırıldığında, işlemci önce PC'yi programın başlangıç adresine ayarlanmakta ve ardından bir komutu işledikçe PC, bir sonraki komutun adresine otomatik olarak geçmektedir.

İşaretçi Registerları (SP, FP, LR): Bu registerlar, verilerin veya program yığınının başlangıç adreslerini ve program akışının kontrolünü sağlamak için kullanılmaktadır. Yığın yönetimi ve fonksiyon çağrıları sırasında kullanılabilmektedirler.

Kaydedici Registerlar (D0-D31): Bu registerlar, 128-bitlik kayan noktalı sayıları veya 64-bitlik çift hassasiyetli tamsayıları depolamak için kullanılmaktadır. İşlemci, kaydedici registerları kullanarak matematiksel işlemler yapabilmektedir.
Sistem Registerları: Bu registerlar, işletim sistemi ve diğer özel amaçlı yazılım bileşenleri tarafından kullanılmaktadır. Bu registerlar, işlemci yapılandırması, bellek yönetimi ve diğer sistem düzeyi işlevlerini kontrol etmek için kullanılmaktadır.

FPSIMD Registerları (V0-V31): Bu registerlar, SIMD (Single Instruction Multiple Data) işlemleri için kullanılmaktadır. Bu registerlar, vektörel matematiksel işlemler için kullanılmakta ve işlemci, bir vektördeki tüm elemanları aynı anda işleyebilmektedir.

Kontrol Registerları: Bu registerlar, çeşitli sistem düzeyi işlevleri için kullanılmaktadır. Örneğin, işlemci modlarını değiştirme, kesmeleri etkinleştirme veya devre dışı bırakma, bellek erişimleri için yetki düzeylerini ayarlama vb.

SVE Registerları: SVE (Scalable Vector Extension) registerları, SIMD işlemleri için kullanılmaktadır, ancak FPSIMD registerlarından farklı olarak, vektör boyutu işlemci tarafından değiştirilebilmekte ve yüksek seviyelerde vektörel işlemler için uyarlanabilmektedir.

Performans Monitör Registerları: Bu registerlar, işlemcinin performansını izlemek için kullanılmaktadır. Bellek erişimleri, dalma tahminleri, döngüler vb. gibi farklı sistem olaylarının sayısını ve süresini ölçebilmektedirler.

Bu register türleri, ARM64 mimarisinin belirli kullanım senaryolarına uyacak şekilde, işlem tasarımı da bu registerların verimli bir şekilde kullanılmasını sağlamak için optimize edilmiştir.

4. SONUÇ

ARM64 tabanlı mikroişlemciler, mobil cihazlardan sunuculara kadar çeşitli cihazlarda yaygın olarak kullanılan güçlü ve enerji verimli işlemcilerdir. Bu işlemcilerin kayıt mimarisi, birçok işlevi yerine getirmek için kullanılan genişletilmiş kayıt seti ile karakterize edilir. Genel amaçlı kayıtlar aritmetik ve mantıksal işlemler için veri depolamak, bellek işlemleri için adresleri tutmak ve işlev argümanlarını ve dönüş değerlerini saklamak için kullanılır. Kayıtların genişliği, erişim hızı, kaynak yönetimi, isimlendirme ve güç verimliliği gibi birçok özelliği vardır. İsimlendirme sistemi, kayıtların kullanımını daha anlaşılır hale getirir. Kayıtların güç verimliliği de özellikle mobil cihazlar gibi pil ömrü kritik cihazlar için önemlidir. Tüm bu özellikler, ARM64 tabanlı mikroişlemcilerin geniş bir uygulama yelpazesinde yaygın olarak kullanılmasını sağlar ve diğer işlemcilere göre avantajlı olmasına yardımcı olur.

Günümüzde ARM64 tabanlı mikroişlemcilerin kullanımı hızla artmakta ve birçok cihazda kullanılmaktadır. Bunun en önemli sebebi bu işlemcilerin enerji verimli olması ve güçlü performans sunmasıdır. ARM64 tabanlı bu işlemcilerin kayıt düzenlemeleri, birçok unsurun yerine getirilmesi için kullanılan kayıt seti ile birlikte karakterize edilmektedir. Bu kayıtlar genel amaçlıdır ve birçok işlevi yerine getirmek için kullanılır. Bunlar; mantıksal ve aritmetik işlemlerin verilerini saklamak, bellek işlemleri için gereken adresleri ve çeşitli argümanları depolamak için kullanılmaktadır.

Kayıtların genişliği, erişim hızı, kaynak yönetimi, adlandırma ve güç yönetimi gibi birçok özelliği vardır. Kayıtların genişliği, işlemcideki veri büyüklüğünü belirlemektedir. Erişim hızı, kayıtların hızlı bir şekilde erişilebilir durumda olmasını sağlamaktadır. Kaynak yönetimi, kayıtların izlenmesi ve kaynakların doğru şekilde korunmasını sağlamak için kullanılmaktadır. Adlandırma sistemi, kayıtların adlandırılmasını sağlar ve kayıtların konuşmalarını daha anlaşılır hale getirir. Güç yönetimi, kayıtların güç kullanımlarını yönetir ve özellikle pil ömrünün kritik olduğu cihazlar için önem taşımaktadır.

ARM64 tabanlı mikroişlemcilerin kullanımı, mobil cihazlar, sunucular, depolama cihazları ve diğer cihazlar gibi geniş bir uygulama yelpazesinde yaygın olarak kullanılmaktadır. Bu işlemcilerin özellikleri, diğer işlemcilere göre daha avantajlı olmaktadır.

KAYNAKLAR

1- Heiser, G. (2011). Hardware-supported virtualization on ARM.

2- https://tr.wikipedia.org/wiki/ARM_mimarisi#:~:text=ARM%20mimarisi%20(orijinal%20ad%C4%B1%20Acorn,dolay%C4%B1%20g%C3%B6m%C3%BCl%C3%BC%20sistemlerde%2C%20ta%C5%9F%C4%B1nabilir%20ayg%C4%B1tlarda

3- Iturbe, X., Venu, B., Ozer, E., Poupat, J. L., Gimenez, G., & Zurek, H. U. (2019). The Arm triple core lock-step (TCLS) processor. *ACM Transactions on Computer Systems (TOCS)*, *36*(3), 1-30.

4- Chiang, H. H., Cheng, H. J., & Hwang, Y. S. (2012, February). Doubling the number of registers on ARM processors. In *2012 16th Workshop on Interaction between Compilers and Computer Architectures (INTERACT)* (pp. 1-8). IEEE.

5- Pan, L., & Liu, X. (2012). The study of target tracking based on ARM embedded platform. *Journal of Computers*, *7*(8), 2015-2023.

6- Teixidó, M., Font, D., Pallejà, T., Tresanchez, M., Nogués, M., & Palacín, J. (2012). An embedded real-time red peach detection system based on an OV7670 camera, ARM cortex-M4 processor and 3D look-up tables. *Sensors*, *12*(10), 14129-14143.

7- Le Gal, B., Leroux, C., & Jego, C. (2014, October). Software polar decoder on an embedded processor. In *2014 IEEE Workshop on Signal Processing Systems (SiPS)* (pp. 1-6). IEEE.

8- https://modexp.wordpress.com/2018/10/30/arm64-assembly/#set

9- https://developer.arm.com/documentation/102404/0201/About-the-Arm-architecture?lang=en

10- Keipert, K., Mitra, G., Sunriyal, V., Leang, S. S., Sosonkina, M., Rendell, A. P., & Gordon, M. S. (2015). Energy-efficient computational chemistry: Comparison of x86 and ARM systems. *Journal of chemical theory and computation*, *11*(11), 5055-5061.

11- Goz, D., Tornatore, L., Taffoni, G., & Murante, G. (2017). Cosmological simulations in exascale era. *arXiv preprint arXiv:1712.00252*.

12- Kalyanasundaram, J., & Simmhan, Y. (2017, December). Arm wrestling with big data: A study of commodity arm64 server for big data workloads. In *2017 IEEE 24th International Conference on High Performance Computing (HiPC)* (pp. 203-212). IEEE.

13- Furber, S. B. (1996). *ARM system Architecture*. Addison-Wesley Longman Publishing Co., Inc..

14- Pulte, C., Flur, S., Deacon, W., French, J., Sarkar, S., & Sewell, P. (2017). Simplifying ARM concurrency: multicopy-atomic axiomatic and operational models for ARMv8. *Proceedings of the ACM on Programming Languages*, *2*(POPL), 1-29.

15- https://eclecticlight.co/2021/06/16/code-in-arm-assembly-registers-explained/

16- https://en.wikipedia.org/wiki/AArch64

17- Seal, D. (Ed.). (2001). *ARM architecture reference manual*. Pearson Education.

18- Programmer's, A. C. A. S. (2015). Guide for ARMv8-A. *ARM*, *97*, 98.

19- https://developer.arm.com/documentation/102374/0101/

20- https://developer.arm.com/documentation/ddi0487/latest/

21- Kalyanasundaram, J., & Simmhan, Y. (2017, December). Arm wrestling with big data: A study of commodity arm64 server for big data workloads. In *2017 IEEE 24th International Conference on High Performance Computing (HiPC)* (pp. 203-212). IEEE.

22- https://eclecticlight.co/2021/07/16/code-in-arm-assembly-bit-operations/

23- https://www.anandtech.com/show/9766/the-apple-ipad-pro-review/3