# Bilgisayar Mimarisi

2023-2024 Bahar Dönemi

Dr. Öğr. Üyesi Nurullah ÖZTÜRK

# CPU-SoC mimarisi

- CPU-SoC (System on Chip) mimarisi, bir bilgisayarın işlemcisi (CPU) ile diğer bileşenlerin (örneğin bellek, grafik işlemcisi, giriş/çıkış birimleri) tek bir entegre devre içinde birleştirildiği bir mimaridir. Bu tasarım, bir bilgisayar veya bir dijital cihazın tüm temel bileşenlerini tek bir çip üzerinde entegre etmeyi amaçlar.

# CPU-SoC mimarisi

Bu mimarinin avantajları şunlardır:

- **Yerden tasarruf:** Tek bir çip içinde birçok bileşenin entegre edilmesi, anakart boyutunu ve dolayısıyla cihazın fiziksel boyutunu azaltabilir.
- **Enerji verimliliği:** Tüm bileşenlerin aynı çipte bulunması, enerji verimliliğini artırabilir. Bileşenler arasındaki iletişim daha hızlı ve daha az enerji tüketir.
- **Daha iyi performans:** Bileşenler arasındaki yakın ilişki, veri transferini hızlandırabilir ve gecikmeleri azaltabilir, bu da genel performansı artırabilir.
- **Daha az gecikme:** Harici bağlantıların yerine, bileşenler arasındaki bağlantılar çok daha kısa olabilir, bu da iletişim gecikmelerini azaltabilir.
- **Daha düşük maliyet:** Tek bir çip üzerindeki tüm bileşenlerin entegrasyonu, üretim maliyetlerini azaltabilir ve sonuç olarak cihazın maliyetini düşürebilir.

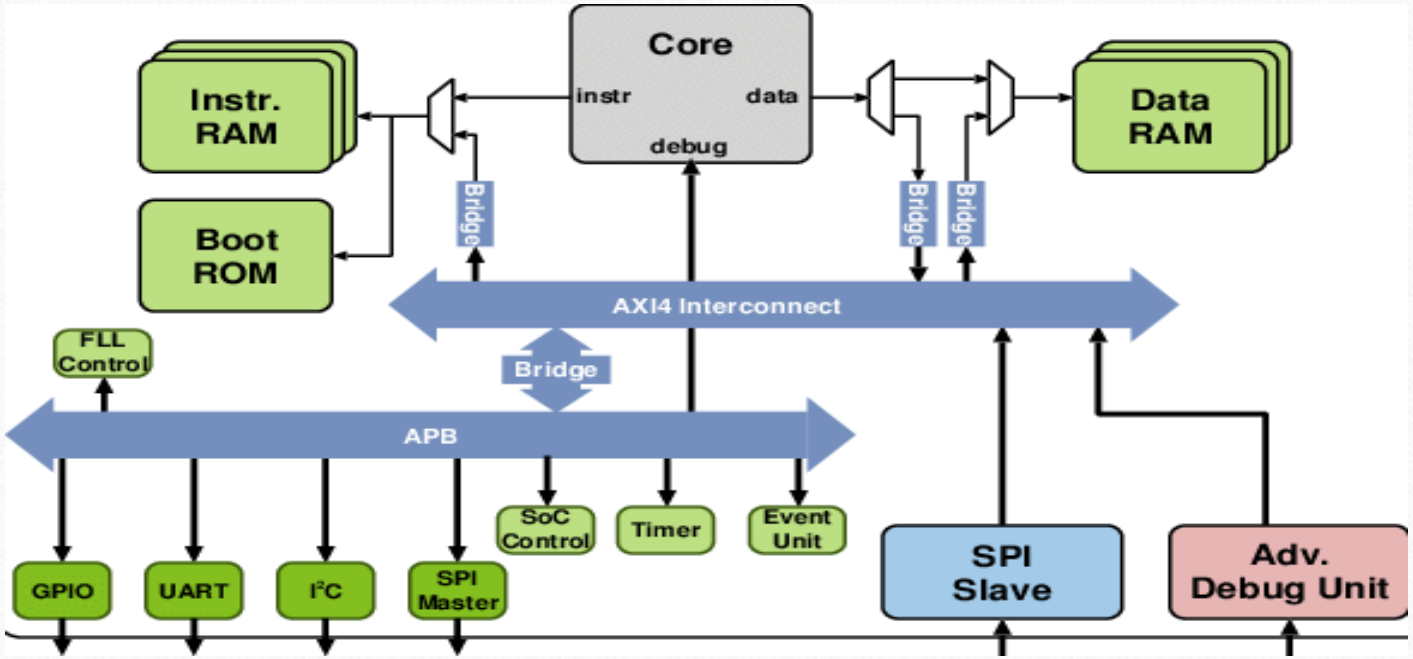
# CPU-SoC mimarisi

- CPU-SoC mimarisi, özellikle mobil cihazlar, akıllı telefonlar, tabletler ve diğer taşınabilir cihazlar için popülerdir, ancak aynı zamanda gömülü sistemler ve diğer türdeki özel amaçlı bilgisayarlar için de kullanılabilir. Bu mimari, daha küçük, daha enerji verimli ve daha verimli cihazlar için temel bir yapı taşı olmuştur.

# Pulpino SoC: ETH Pulp Team

- Pulpino SoC, ETH Zurich'deki Pulp Takımı tarafından geliştirilen bir System on Chip (SoC) tasarımıdır. Pulpino, açık kaynaklı bir RISC-V işlemci mimarisine dayalıdır ve mikrodenetleyici uygulamaları için tasarlanmıştır. ETH Pulp Takımı, özellikle düşük güç tüketimi, yüksek performans ve ölçeklenebilirlik gibi özelliklere odaklanarak, gömülü sistemler için enerji verimli ve esnek bir işlemci tasarımı sunmayı amaçlamaktadır.

# Pulpino SoC: ETH Pulp Team



# Pulpino SoC: ETH Pulp Team

Pulpino SoC'nin temel bileşenleri şunlardır:

- **RISC-V İşlemcisi:** Pulpino, açık kaynaklı RISC-V işlemci mimarisine dayanan bir mikrodenetleyici içerir. RISC-V, basit, temiz bir tasarıma sahip açık kaynaklı bir işlemci mimarisidir.
- **Bellek Alt Sistemi:** Pulpino SoC, program kodunun ve verilerin depolanması için bir bellek alt sistemi içerir. Bu, işlemcinin ihtiyaç duyduğu talimatlar ve veriler için depolama sağlar.
- **Giriş/Çıkış Birimleri:** Pulpino SoC, harici cihazlarla iletişim kurmak için giriş/çıkış birimlerine sahiptir. Bu, sensörler, ekranlar, depolama birimleri ve diğer harici cihazlarla etkileşim sağlar.
- **Timers ve Saatler:** Pulpino, zamanlama işlevselliği için dahili zamanlayıcılar ve saatler içerir. Bu, zamanlayıcı tabanlı işlemler için zamanlama ve senkronizasyon sağlar.
- **Enerji Yönetimi:** Pulpino SoC, enerji tasarrufu sağlamak için çeşitli enerji yönetimi özelliklerine sahiptir. Bu, pil ömrünü uzatmak veya enerji tüketimini optimize etmek için güç tüketimini ayarlamayı sağlar.

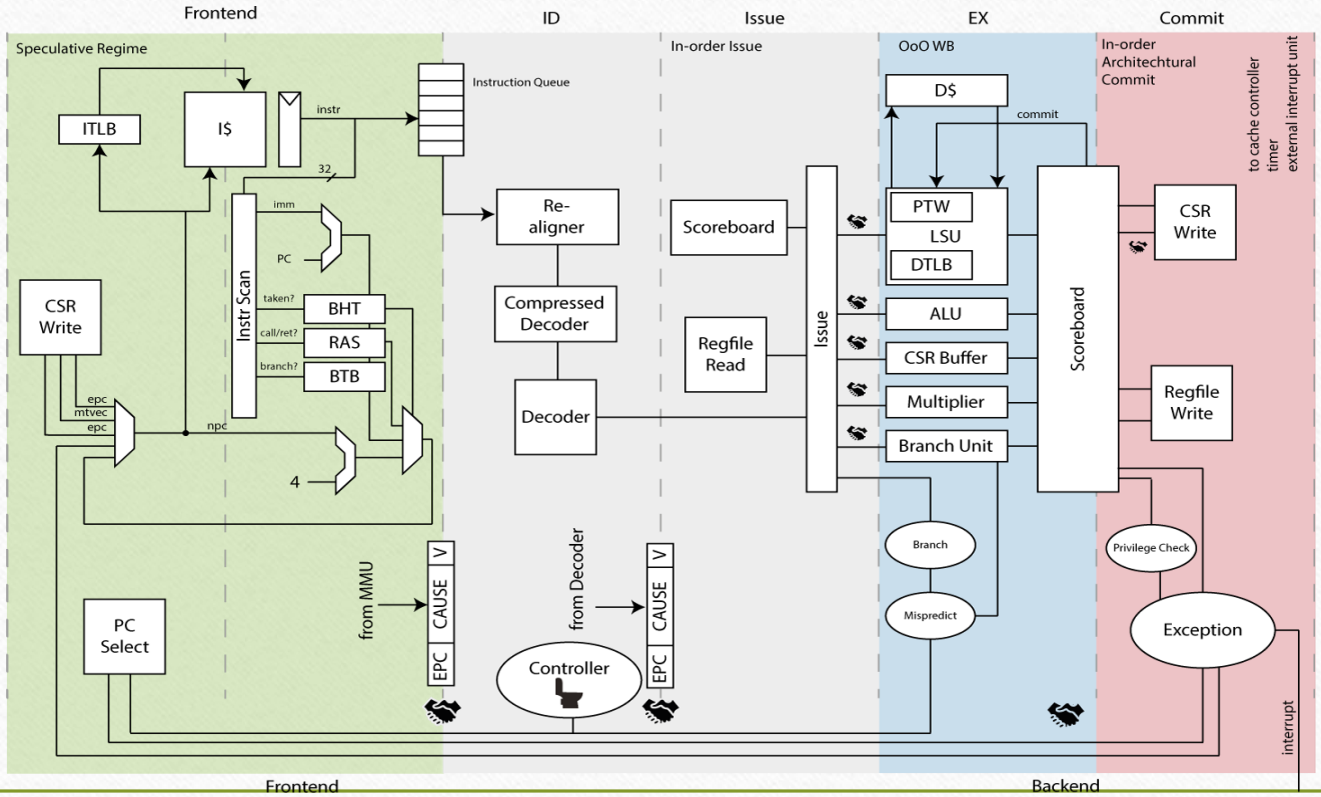
# Pulpino SoC: ETH Pulp Team

ETH Pulp Takımı, Pulpino SoC'nin gelişimini açık kaynaklı olarak sunmuştur, bu da diğer geliştiricilerin tasarımı incelemesini, değiştirmesini ve kullanmasını sağlar. Bu, Pulpino'nun esnekliğini artırır ve geniş bir topluluğun katkıda bulunmasını teşvik eder. Pulpino, özellikle IoT (Nesnelerin İnterneti) cihazları, giyilebilir teknolojiler ve diğer gömülü sistemler için enerji verimli bir işlemci çözümü olarak popülerdir.

# CVA6 RISC-V CPU: OPENHWGROUP

- CVA6, RISC-V işlemci mimarisi üzerine inşa edilmiş bir mikrodenetleyici çekirdeğidir ve OpenHW Group tarafından geliştirilmiştir. OpenHW Group, açık kaynaklı donanım tasarımı topluluğudur ve üyelerine açık kaynaklı mikrodenetleyici ve diğer sistem tasarımı bileşenlerine erişim sağlar.
- CVA6, RISC-V Vekil B (RV32IMC) komut setiyle uyumlu bir çekirdek olarak tasarlanmıştır. RV32IMC, 32-bit RISC-V komut setine dayanan bir mimaridir ve çekirdeğin temel özelliklerini oluşturur. Bu komut seti, integer işlemleri (I), çoğaltma/ölçekleme işlemleri (M), sıkıştırılmış talimatlar (C) ve atomik işlemler (A) gibi temel hesaplama ve bellek erişimi işlevselliğini içerir.

# CVA6 RISC-V CPU: OPENHWGROUP



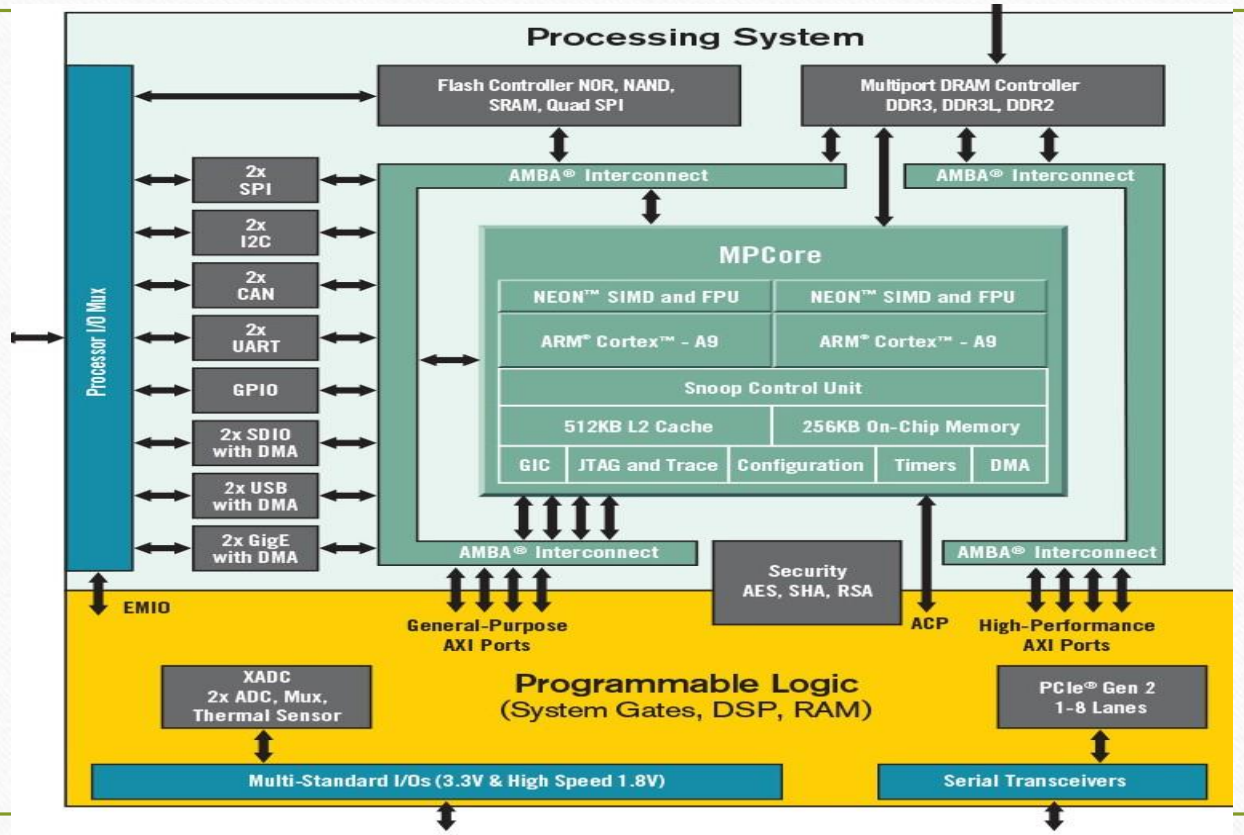
# CVA6 RISC-V CPU: OPENHWGROUP

- CVA6, düşük güç tüketimi, yüksek performans ve ölçeklenebilirlik gibi özellikler sunmayı amaçlar. Ayrıca, açık kaynaklı bir tasarım olduğu için, topluluk tarafından incelenebilir, değiştirilebilir ve kullanılabilir. Bu, CVA6'nın farklı uygulama alanlarına ve ihtiyaçlara göre özelleştirilebilir olmasını sağlar.
- OpenHW Group, CVA6 ve diğer açık kaynaklı donanım tasarımları üzerinde çalışarak, endüstride açık kaynaklı donanımın benimsenmesini teşvik etmeyi ve donanım tasarımının erişilebilirliğini artırmayı amaçlar. Bu, donanım geliştiricilerinin daha kolay ve hızlı bir şekilde ürünler geliştirmesini sağlar ve endüstride inovasyonu teşvik eder.

# XILINX ZYNQ SoC

- Xilinx Zynq SoC (System on Chip), Xilinx tarafından geliştirilen ve FPGA (Field-Programmable Gate Array - Alan Programlanabilir Kapı Dizisi) teknolojisi ile ARM işlemci çekirdeklerinin birleştirildiği bir çipli bir platformdur. Bu platform, hem programlanabilir mantık hem de işlemci gücünü bir araya getirerek yüksek performanslı ve esnek bir sistem tasarlama imkanı sağlar.
- Zynq SoC, iki temel bileşeni bir araya getirir:

# XILINX ZYNQ SoC



# XILINX ZYNQ SoC

- **Programlanabilir Mantık (FPGA):** Zynq SoC, FPGA'nın sunduğu esneklik ve özelleştirme yeteneklerinden yararlanır. FPGA, kullanıcıların donanımı özelleştirmesine ve işlevselliği yazılım tabanlı olarak güncellemesine izin verir. Bu, farklı endüstriyel uygulamalar için optimize edilmiş donanım hızlandırmaları sağlar.
- **ARM İşlemci Çekirdekleri:** Zynq SoC ayrıca ARM işlemci çekirdeklerini içerir. Bu işlemciler, genel amaçlı işlemleri yönetmek ve yazılım tabanlı uygulamaları çalıştırmak için kullanılır. ARM çekirdekleri, birden çok işlemci çekirdeği, grafik işlemcileri, video işleme birimleri ve diğer işlevsel birimlerle birleştirilebilir, böylece geniş bir uygulama yelpazesi için uygun bir platform sağlar.

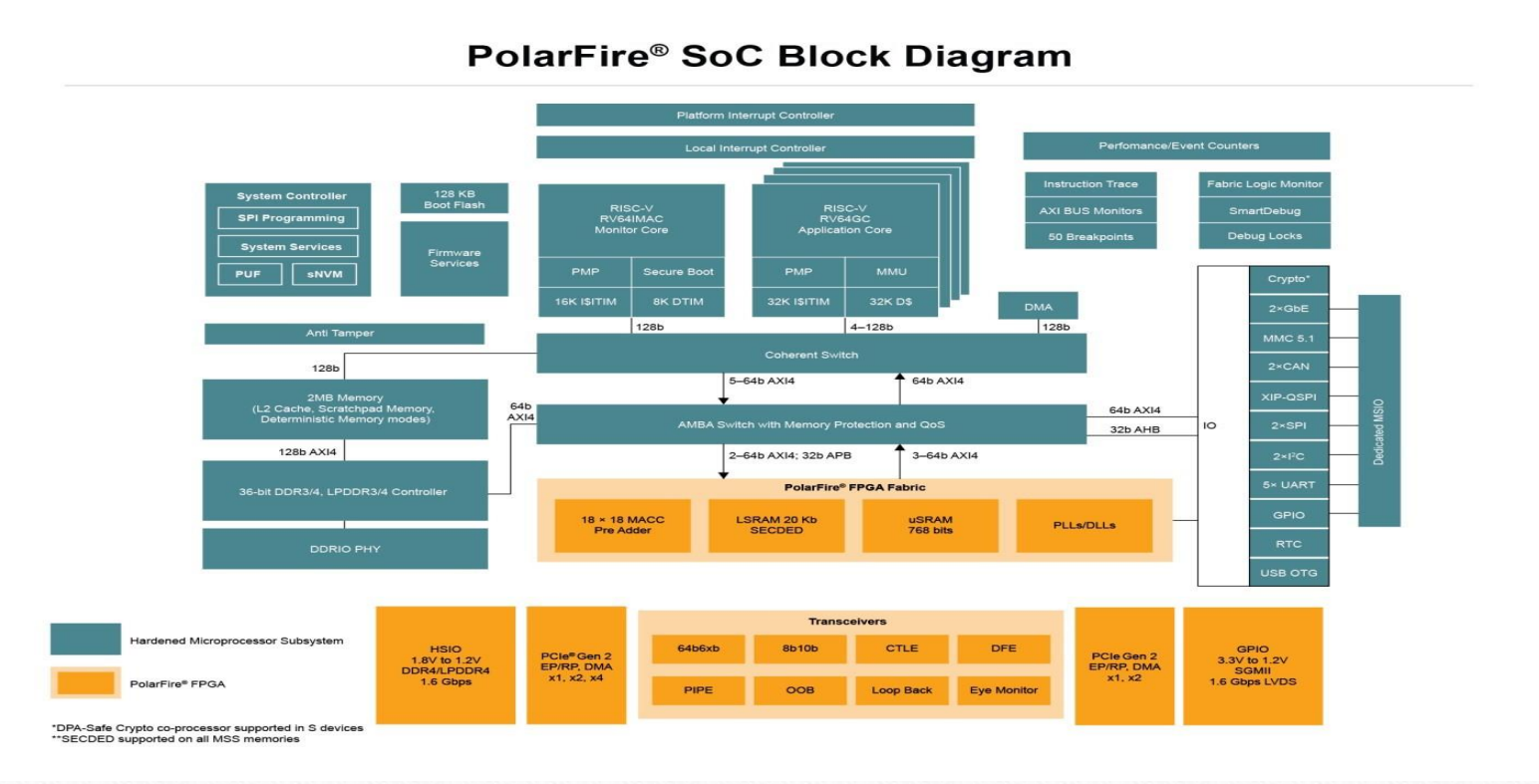
# XILINX ZYNQ SoC

- Zynq SoC'ler genellikle gömülü sistemler, ağ ekipmanları, endüstriyel otomasyon, otomotiv uygulamaları, tıbbi cihazlar ve diğer yüksek performanslı, düşük güç tüketen uygulamalar için kullanılır. Bu tür uygulamalarda, hem donanım hızlandırması hem de genel amaçlı işlemci gücünün birleşimi, performansı artırırken esneklik ve özelleştirme imkanı sağlar.

# MICROCHIP POLARFIRE SoC

- Microchip PolarFire SoC, güçlü FPGA ve işlemci gücünün birleştirildiği bir System on Chip (SoC) platformudur. Bu platform, endüstriyel IoT (Nesnelerin İnterneti), otomotiv, telekomünikasyon, savunma ve diğer benzeri uygulamalar için yüksek güvenilirlik, düşük güç tüketimi ve yüksek performans sunmayı amaçlar.

# MICROCHIP POLARFIRE SoC



# MICROCHIP POLARFIRE SoC

PolarFire SoC'nin temel özellikleri şunlardır:

- **FPGA:** PolarFire SoC, Microchip'in PolarFire FPGA serisinden güçlü bir FPGA içerir. FPGA, kullanıcıların özelleştirilmiş donanım tasarlamasına ve uygulamasına olanak tanır. PolarFire FPGA'lar, düşük güç tüketimi, yüksek güvenilirlik ve yüksek bant genişliği gibi özellikler sunar.
- **RISC-V İşlemci Çekirdekleri:** PolarFire SoC, RISC-V işlemci çekirdeklerini içerir. RISC-V, açık kaynaklı bir işlemci mimarisidir ve PolarFire SoC'deki işlemciler genellikle eklenebilir işlemciler veya gömülü kontrol birimleri olarak kullanılır. RISC-V'nin açık kaynaklı doğası, donanım ve yazılım geliştiricilerinin platform üzerinde esneklik ve özelleştirme sağlar.
- **Düşük Güç Tüketimi:** PolarFire SoC, düşük güç tüketimiyle dikkat çeker. Bu özellik, taşınabilir cihazlar, pil destekli uygulamalar ve diğer enerji duyarlı uygulamalar için idealdir.

# MICROCHIP POLARFIRE SoC

- **Gelişmiş Güvenlik:** Platform, güvenlik önlemleri içerir ve endüstriyel ve askeri uygulamalar gibi güvenlik gereksinimleri yüksek olan alanlar için uygundur.
- **Esneklik ve Özelleştirme:** FPGA'nın esnek doğası ve açık kaynaklı RISC-V işlemcileri, kullanıcılara donanım ve yazılımı ihtiyaçlarına göre özelleştirme imkanı sunar.

PolarFire SoC, çeşitli endüstriyel ve askeri uygulamalarda, ağ ekipmanları, veri merkezi altyapısı, video işleme sistemleri, otomotiv elektroniği ve diğerleri gibi geniş bir uygulama yelpazesinde kullanılır. Bu tür uygulamalarda, yüksek performans, düşük güç tüketimi ve güvenilirlik gibi özellikler kritiktir ve PolarFire SoC bu gereksinimleri karşılamak için tasarlanmıştır.

# Sayı Sistemleri

10'luk Taban

- 20654 | $(20654)_{10}$
- 2*10^4 + 0*10^3 + 6*10^2 + 5*10^1 + 4*10^0 =
- 20000 + 0 + 600 + 50 + 4

Ondalıklı sayılar ve genel kuram:

$$10^5 a_5 + 10^4 a_4 + 10^3 a_3 + 10^2 a_2 + 10^1 a_1 + 10^0 a_0 + 10^{-1} a_{-1} + 10^{-2} a_{-2} + 10^{-3} a_{-3}$$

# Sayı Sistemleri

2'lik Taban

- $(11001)_2$
- 1*2^4 + 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0 =
- 16 + 8 + 0 + 0 + 1
- $(25)_{10}$

# Sayı Sistemleri

2'lik Taban

$(11010.11)_2 = (?)_{10}$

$$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = 26.75$$

5'lik Taban

$$(4021.2)_5 = 4 \times 5^3 + 0 \times 5^2 + 2 \times 5^1 + 1 \times 5^0 + 2 \times 5^{-1} = (511.4)_{10}$$

# Sayı Sistemleri

8'lik Taban

$$(127.4)_8 = 1 \times 8^2 + 2 \times 8^1 + 7 \times 8^0 + 4 \times 8^{-1} = (87.5)_{10}$$

16'lik Taban

$$(B65F)_{16} = 11 \times 16^3 + 6 \times 16^2 + 5 \times 16^1 + 15 \times 16^0 = (46{,}687)_{10}$$

Hex karakterler : 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Ornek Gösterim

: 0x35 → 0011 0101 (8-bit = 1 byte)

: 0x13AF → 0001 0011 1010 1111 (16-bit / 1 word)

: 0xA7D394FB→ 1010 0111 1101 0011 1001 0100 1111 1011 (32-bit / 1 word)

# Taban Dönüşümü

$(25)_{10} = (?)_2$

| Sayı | Bölüm | Kalan | Bit Değer | Bit Sıra |
|---|---|---|---|---|
| 25 | 12 | 1 | 1 | 0 |
| 12 | 6 | 0 | 0 | 1 |
| 6 | 3 | 0 | 0 | 2 |
| 3 | 1 | 1 | 1 | 3 |
| 1 | 0 | 1 | 1 | 4 |

11001

# Taban Dönüşümü

$0.6875_{10} = (?)_2$

| Sayı | Çarpım | Elde | Bit Değer | Bit Sıra |
|---|---|---|---|---|
| 0.6875 | 0.375 | 1 | 1 | -1 |
| 0.375 | 0.75 | 0 | 0 | -2 |
| 0.75 | 0.5 | 1 | 1 | -3 |
| 0.5 | 0 | 1 | 1 | -4 |

$(0.1011)_2$

# Taban Dönüşümü

- $0.513_{10} = (?)_2$

| Sayı | Çarpım | Elde | Bit Değer | Bit Sıra |
|---|---|---|---|---|
| 0.513 | 0.026 | 1 | 1 | -1 |
| 0.026 | 0.052 | 0 | 0 | -2 |
| 0.052 | 0.104 | 0 | 0 | -3 |
| 0.104 | 0.208 | 0 | 0 | -4 |
| 0.208 | 0.416 | 0 | 0 | -5 |
| 0.416 | 0.832 | 0 | 0 | -6 |
| 0.832 | 0.664 | 1 | 1 | -7 |
| 0.664 | 0.328 | 1 | 1 | -8 |

Eğer ondalıklı kısım için 8-bit ayrıldıysa 8 kere iterasyon devam ettirilir.

$$(0.513)_{10} = (0.10000011)_2$$

$$(0.10000011)_2 = (0.51171875)_{10}$$

# Taban Tümleme

Çıkarma işlemini basitleştirme amacıyla kullanılmaktadır

Çıkarma işlemi, aslında toplama işleminde 2. toplanan sayının işaretinin değiştirilip toplanması ile meydana gelir

Tümleme işlemi 10'luk tabanda sayılar için 9 (10-1) ve 10'a tümleyen şeklinde tanımlanmıştır

# Taban Tümleme

- Örn: 3: 9'a tümleyen → 6, 10'a tümleyen → 7
- Örn: 9: 9'a tümleyen → 0, 10'a tümleyen → 1
- Örn: 7: 9'a tümleyen → 2, 10'a tümleyen → 3

3+9 = 12
3-9 = -6
3-9 = 3+1-10 = -6

9+7 = 16
9-7 = 2
9-7 = 9+3-10 = 2

| | | | | | |
|---|---|---|---|---|---|
| 3 | 3 | 4 | 9 | 9 | 12 |
| − 9 | + 1 | − 10 | − 7 | + 3 | − 10 |
| -6 | 4 | -6 | 2 | 12 | 2 |

Kural: Eğer 10'a tümleyenle topladığında elde yoksa sonuç kalan sayının 10'a tümleyeninin eksi işaretlisidir
Kural: Eğer 10'a tümleyenle topladığında elde varsa sonuç elde hariç kalan sayıdır

# Taban Dönüşümü

- Tümleme işlemi 2'lik tabanda sayılar için 1 (2-1) ve 2'ye tümleyen şeklinde tanımlanmıştır

Örn: 0: 1'e tümleyen → 1, 2'ye tümleyen → 0

Örn: 1: 1'e tümleyen → 0, 2'ye tümleyen → 1

Örn: 1101: 1'e tümleyen → 0010, 2'ye tümleyen → 0011

$$\begin{array}{r} 0011 \\ +\ 1001 \\ \hline 1100 \end{array} \qquad \begin{array}{r} 0011 \\ -\ 1001 \\ \hline 11010 \end{array} \qquad \begin{array}{r} 0011 \\ +\ 0111 \\ \hline 1010 \end{array}$$

1'e tümleyeni 0101 → 2'ye tümleyeni 0110 → Eksi işaretlisi -6

$$\begin{array}{r} 1001 \\ +\ 0111 \\ \hline 10000 \end{array} \qquad \begin{array}{r} 1001 \\ -\ 0111 \\ \hline 0010 \end{array} \qquad \begin{array}{r} 1001 \\ +\ 1001 \\ \hline 10010 \end{array}$$

→ Elde var → Sonuç elde hariç kalan kısım → 0010 = 2

# İşaretli 2'li Sayılar

- 10'luk sistemde ve matematikte işaretli (eksi) sayılar başlarında "-" sembolü kullanır.

-9, -12, -3.14

- 2'lik sistemde en anlamlı bit (MSB) 1 ise sayı eksi, 0 ise artı işaretlidir.
- 2'lik sistemde işaretli sayıları göstermek için 3 farklı yöntem:
  - -İşaretli Büyüklük (Signed Magnitude)
  - -İşaretli 1'e Tümleyen (Signed 1's Complement)
  - -İşaretli 2'ye Tümleyen (Signed 2's Complement)

01001: 9

11001: ???

# İşaretli 2'li Sayılar

- 11001 = ???
- İşaretli Büyüklük: En anlamlı bit 1 ise, geri kalan bit dizisi işaretsiz olarak değeri hesaplanır ve eksi işaret eklenir:
- 11001: -(1001) → -9
- İşaretli 1'e Tümleyen: En anlamlı bit 1 ise, geri kalan bit dizisinin1'e tümleyeni alınır ve eksi işaret eklenir:
- 11001: -(0110) → -6
- İşaretli 2'ye Tümleyen: En anlamlı bit 1 ise, geri kalan bit dizisinin 2'ye tümleyeni alınır ve eksi işaret eklenir:
- 11001: -(0111) → -7

- NOT: Günümüz bilgisayarlarda İşaretli2'ye Tümleyen (Signed 2's Complement) metodu kullanılmaktadır

# İşaretli 2'li Sayılar

Günümüz bilgisayarları neden 2'ye tümleyen metodunu kullanıyorlar?

5 bit ile İşaretli Büyüklük yöntemiyle gösterilebilecek sayılar: -15,-14,...,-1,-0,+0,1,...,15
5 bit ile İşaretli 1'e Tümleyen yöntemiyle gösterilebilecek sayılar: -15,-14,...,-1,-0,+0,1,...,15
5 bit ile İşaretli 2'ye Tümleyen yöntemiyle gösterilebilecek sayılar: -16,-15,...,-1,0,1,...,15

Toplama/Çıkarma İşlemleri:

| | | | |
|---|---|---|---|
| + 6 | 00000110 | − 6 | 11111010 |
| +13 | 00001101 | +13 | 00001101 |
| +19 | 00010011 | + 7 | 00000111 |
| + 6 | 00000110 | − 6 | 11111010 |
| −13 | 11110011 | −13 | 11110011 |
| − 7 | 11111001 | −19 | 11101101 |

# Taşma

- Bilgisayarlarda (ya da 2'li aritmetik yapan herhangi bir sayısal devrede) aritmetik işlemlerin yapıldığı birimlerin sınırlı alana sahip olduğunu hiçbir zaman unutmamak gerekir.

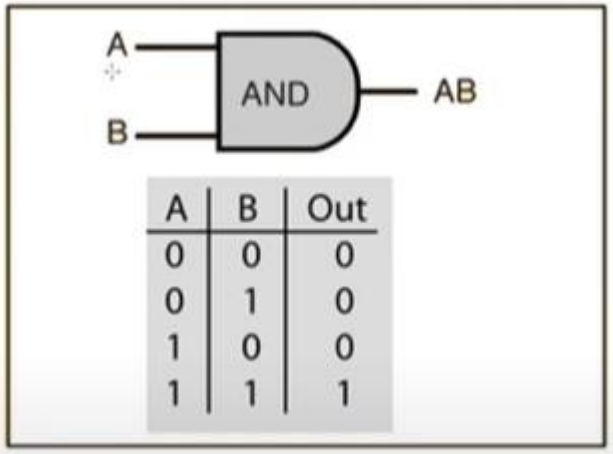
Örnek olarak 4-bit büyüklüğünde sayılarla işlem yaptığımızı düşünelim → -8,-7,...,6,7

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1001: | - 7 | 0111: | +7 | 0111: | +7 | 1101: | -3 |
| + 0111: | +7 | − 1011: | -5 | + 0101: | +5 | + 1010: | -6 |
| 10000: | 0 | ????: | +12 | 1100: | -4 | 10111: | +7 |

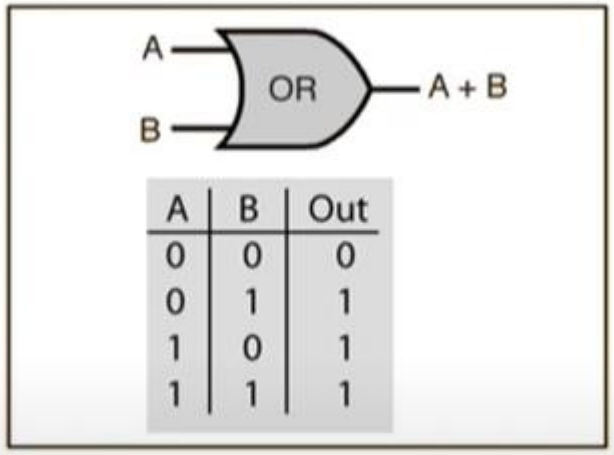
# Taşma

| | | |
|---|---|---|
| unsigned char | : 8-bit işaretsiz sayılar | → 0,1,...,255 |
| signed char | : 8-bit işaretli sayılar | → -128,-127,...,126,127 (-2^7,...,2^7-1) |
| int | : 32-bit işaretli sayılar | → -2_147_483_648,..., 2_147_483_647 (-2^31,...,2^31-1) |
| long | : 64-bit işaretli sayılar | → (-2^63,...,2^63-1) |
| float | : 32-bit kayan sayılar | → (-1.2E-38 to 3.4E+38) |
| double | : 64-bit kayan sayılar | → (-2.3E-308 to 1.7E+308) |

# Temel Mantık Kapıları

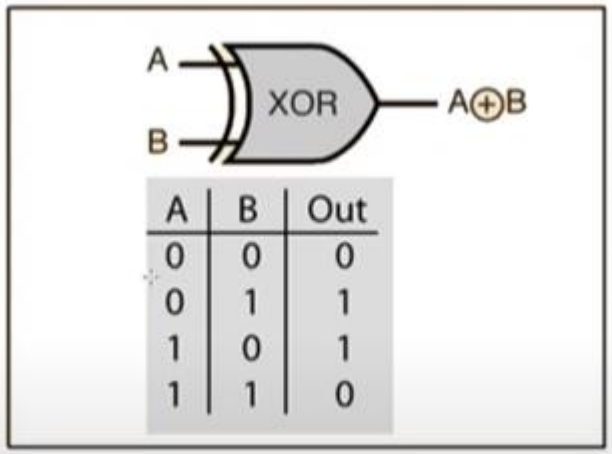



| A | B | Out |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

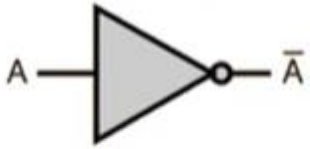



| A | B | Out |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

# Temel Mantık Kapıları



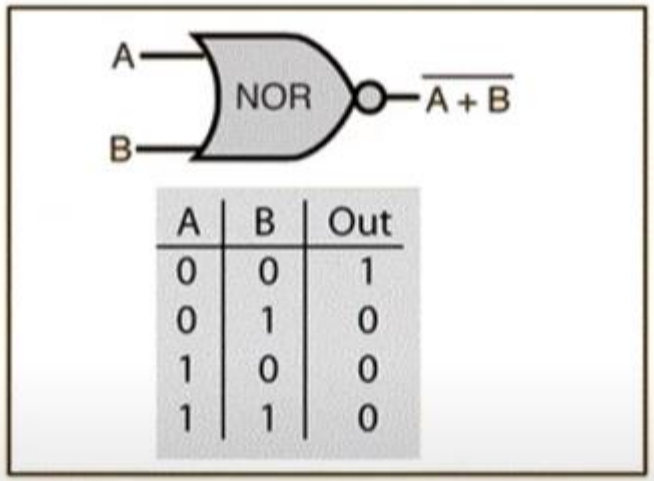

| A | B | Out |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



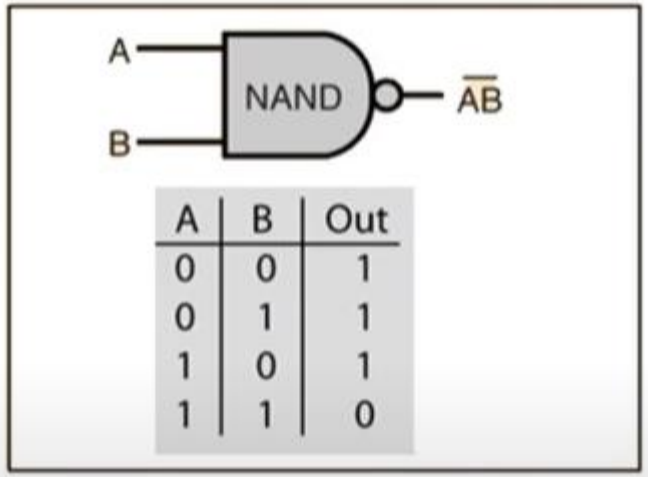

Inverting Buffer

| In | Out |
|---|---|
| 0 | 1 |
| 1 | 0 |

# Temel Mantık Kapıları




| A | B | Out |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |




| A | B | Out |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

# Boole Cebri

1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $A \cdot A = A$
8. $A \cdot \overline{A} = 0$
9. $\overline{\overline{A}} = A$
10. $A + AB = A$
11. $A + \overline{A}B = A + B$
12. $(A + B)(A + C) = A + BC$

# DeMorgan Teoremi

- X ve Y'nin değili
  - $(XY)' = X' + Y'$
- X veya Y nin değili
  - $(X+Y)' = X'Y'$

- $F=AB+C'D'+B'D$
- $F'=(A'+B')(C+D)(B+D')$