



Digitális Áramkörök

(Villamosmérnök BSc /
Mechatronikai mérnök MSc)

13. hét: Kombinációs, sorrendi hálózatok
megvalósítása LSI/MSI áramkörökkel (PAL, PLA,
PROM, CPLD), VLSI (FPGA) áramkörökkel

Előadó: Dr. Vörösházi Zsolt

voroshazi.zsolt@virt.uni-pannon.hu

Kapcsolódó jegyzet, segédanyag:

- <http://www.virt.uni-pannon.hu>
→ Oktatás → Tantárgyak → Digitális
Áramkörök (Villamosmérnök BSc /
Mechatronikai mérnök BSc/MSc).
- Fóliák, óravázlatok (.ppt)
- Feltöltésük folyamatosan

Kombinációs hálózatok / szekvenciális hálózatok (LSI/MSI):

■ Programozható logikai áramkörök_(PLD) alkalmazása K.H., illetve S.H. tervezésére:

- ☐ PAL (Programmable AND Logic)
- ☐ PLA (Programmable Logic Array)
- ☐ PROM (Programmable OR – Read Only memory)
- ☐ GAL (Generic Array Logic)



- ☐ **CPLD** (Complex Programmable Gate Array)



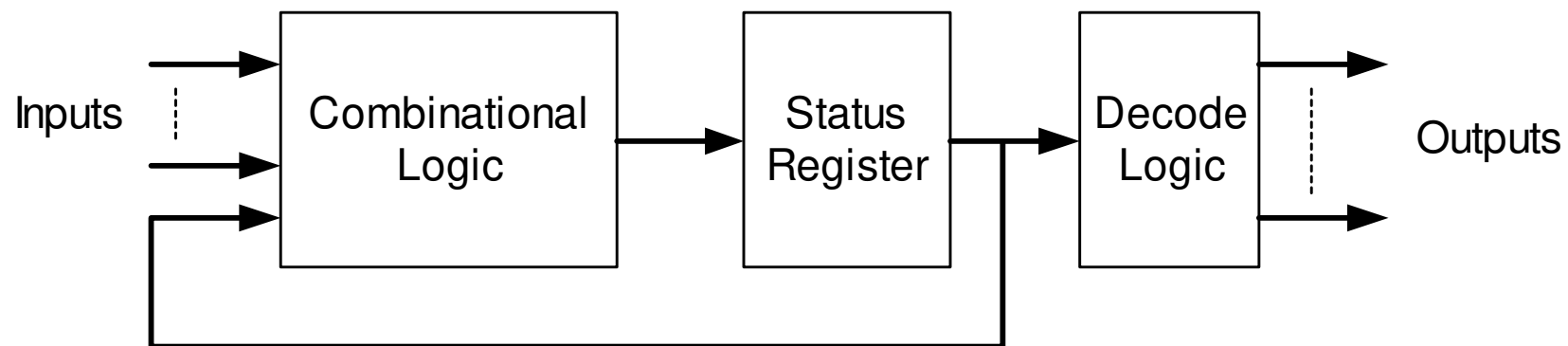
- ☐ **FPGA** (Field Programmable Gate Array)



Programozható logikai eszközök (PLD-k)

Állapotgép FSM tervezés tulajdonságai

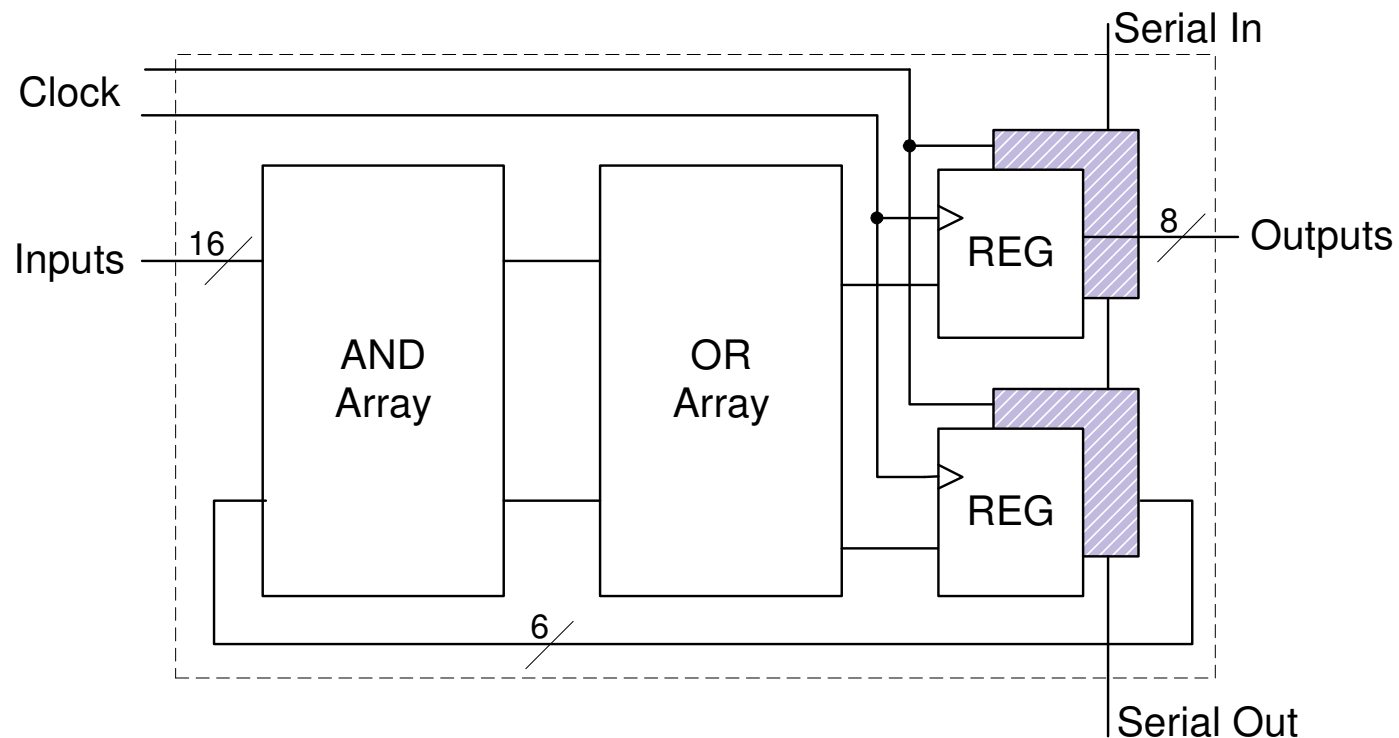
- Két kombinációs logikai hálózatról és egy regiszterből áll
- Tervezés során az állapot-átmeneteket vesszük figyelembe, **DE**
- Hibavalószínűség nagy,
- Szimulációs eszközök (CAD Tools) hiánya,
- Hibák lehetségesek a prototípus fejlesztése során is,
- Könnyen konfigurálható / flexibilis eszközök szükségesek
→ mindezek használunk programozható alkatrészeket



Field Programmable Logic Sequencer (FPLS) – Logikai sorrendvezérlő

- Egy vezérlő egység Next-State (=K.H.) logikai blokkjának megvalósítása a tervező, gyártó feladata. Általában változtatható logikát használnak a függvények megvalósításánál.
- A felhasználó által programozható logikai sorrendvezérlő (**Field Programmable Logic Sequencer**) programozható alkatrészekből építhető fel, amelyek a következőkben részletesen ismertetésre kerülnek.

Field Programmable Logic Sequencer (FPLS) – Logikai sorrendvezérlő

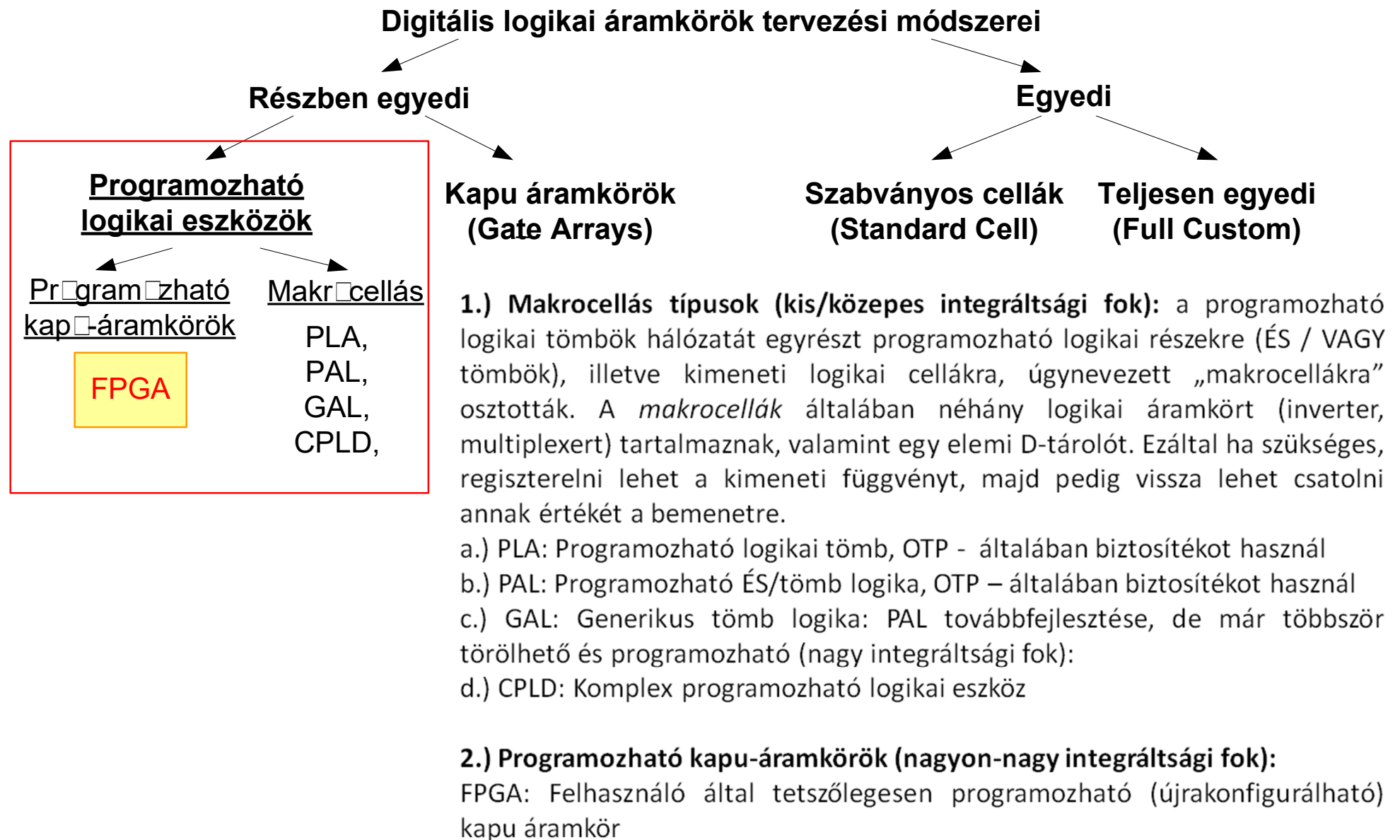


Működése:

- Normál (D-FF),
- Debug (Shift Reg.)

A logikai sorrendvezérlőnek 16 külső bemeneti vonala van, 1 RESET és 1 kimenet-engedélyező vonala, ill. 8 kimeneti vonala. A regiszterek egy-egy állapotot tárolnak, amelyek az órajel hatására a kimenetre íródnak, vagy a 6 belső, *visszacsatoló* vonalon keresztül visszacsatolódnak. A Next-State ill. a kimeneti szintek meghatározásánál programozható AND/OR tömböket használnak.

Tervezési módszerek



PLD – Programozható logikai eszközök

- A Programozható logikai áramköröket (*PLD: Programmable Logic Devices*) általánosan a kombinációs logikai hálózatok és sorrendi hálózatok tervezésére használhatjuk. Napjainkban ezek *VLSI (Very-Large Scaling Integrated IC)* típusú alkatrészecskék:
 - Több tíz-millió logikai kaput, vagy tranzisztort jelenthet
- Azonban míg a hagyományos kombinációs logikai hálózatok dedikált összeköttetésekkel, illetve kötött funkcióval (kimeneti függvény) rendelkeznek, addig a programozható logikai eszközökben pontosan ezek változtathatók, az alábbi lehetséges módokon:
 - A felhasználó által *egyszer programozható* / konfigurálható logikai eszközök (**OTP**: One Time Programmable), amelynél a gyártás során nem definiált funkció egyszer megváltoztatható (ilyenek pl. a korai PAL, PLA eszközök),
 - *Többször, akár tetszőleges módon* programozható logikai eszközök = *újraconfigurálható* (ilyenek pl. a korábbi GAL, vagy a mai modern CPLD, **FPGA** eszközök)

PLD-k két fő típusa:

■ 1.) **Makrocellás** PLD-k: (Programmable Logic Devices):

- ☐ PLA
- ☐ PAL
- ☐ GAL
- ☐ **CPLD**

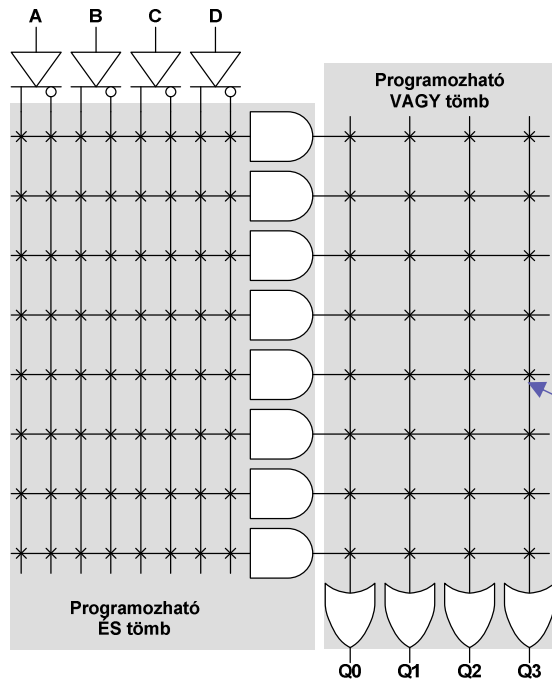
■ 2.) **FPGA** (Field Programmable Gate Array): Programozható Gate Array áramkörök

- ☐ **XILINX** (Spartan, Virtex, Kintex, Artix) ~ 53 % !
- ☐ **Altera/Intel-FPGA** (Stratix, Arria, Cyclone), ~ 36 %
- ☐ MicroSemi-Actel (pl. úrkutatásban),
- ☐ Lattice sorozatok,
- ☐ QuickLogic,
- ☐ További kisebb gyártók termékei.

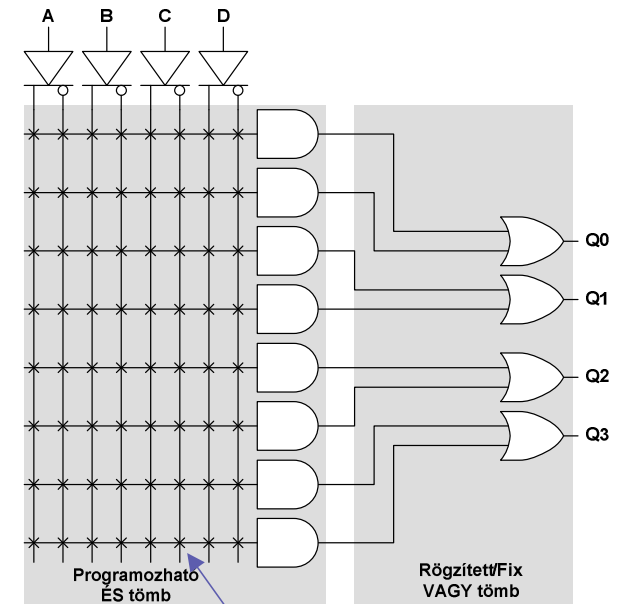


Makrocellás PLD-k

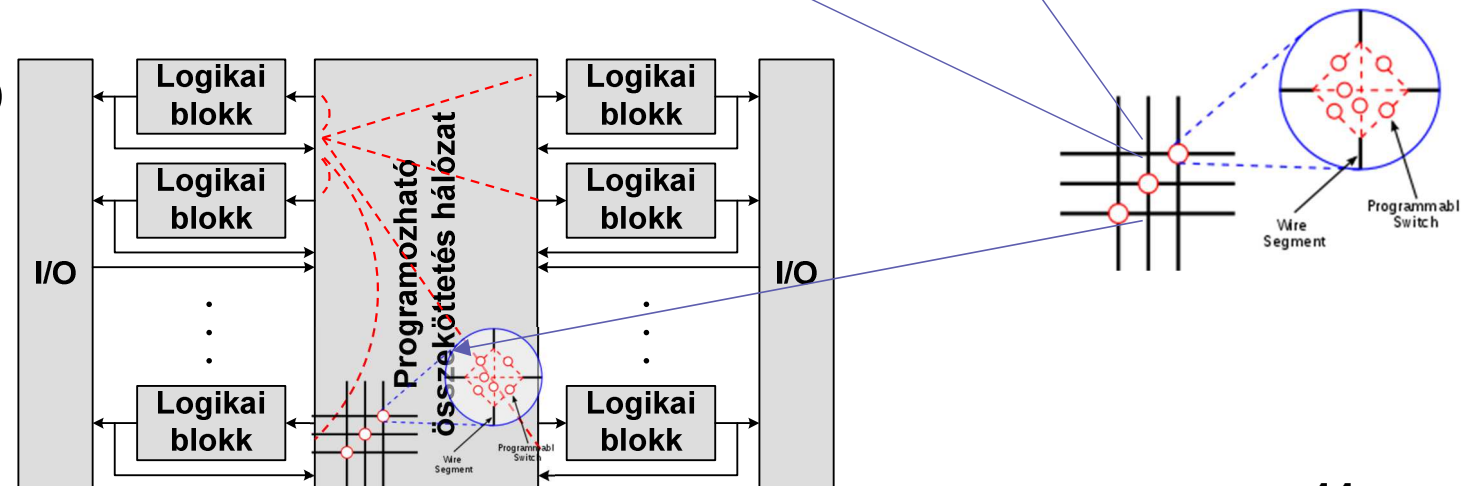
PLA



PAL

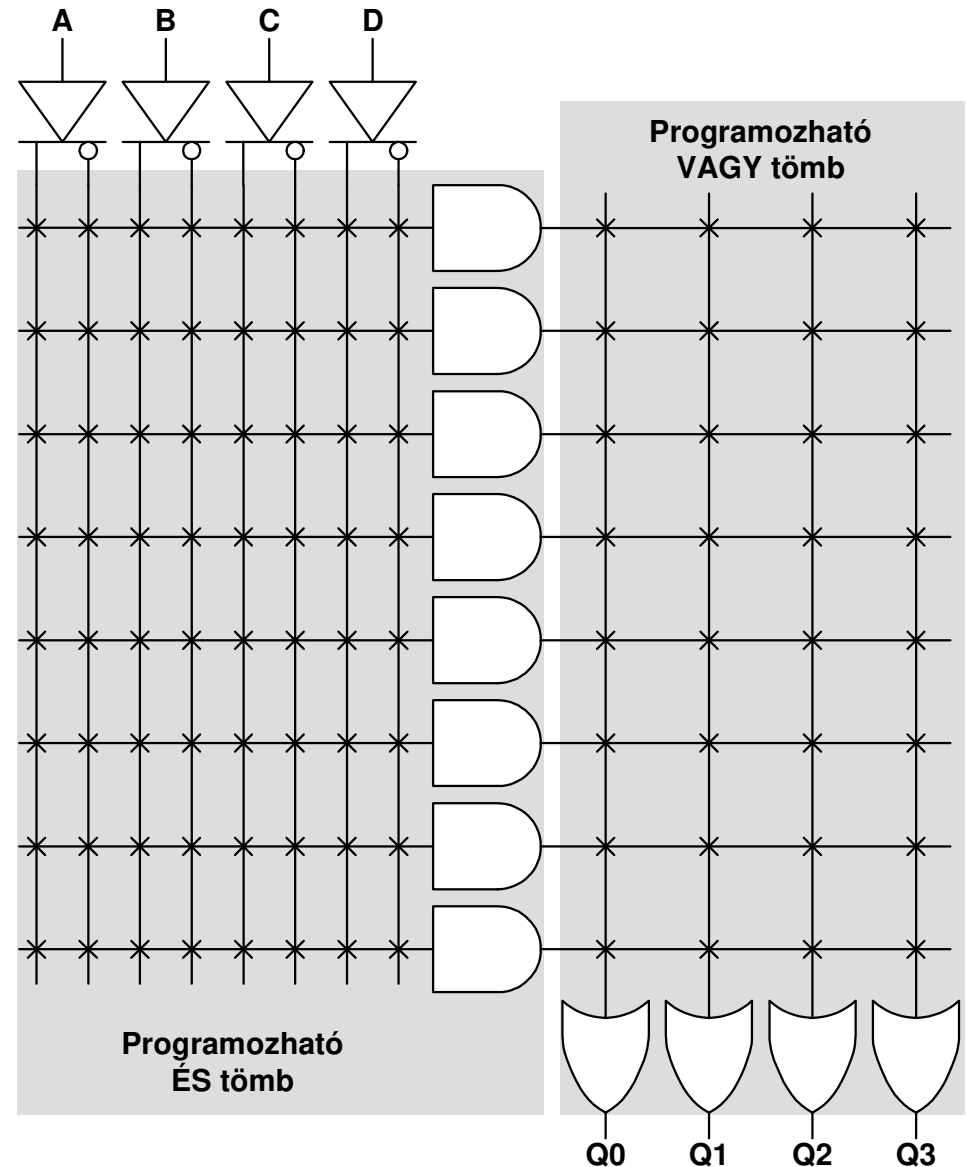


CPLD



Programmable Logic Array (PLA)

- Mindkét része (AND, OR) programozható
- Bármely kombinációja az AND / OR-nak előállítható
- Mintermek OR kapcsolata (DNF)
- Programozható kapcsolók a horizontális/ vertikális vonalak metszésében
- Q_n Kimeneteken D tárolók! (visszacsat. a bemenetekre)



PLA

- 1970-ben, a TI (Texas Instruments) által kifejlesztett eszköz mindkét részhálózata (ÉS, illetve VAGY tömb) programozható összeköttetéseket tartalmazott, amelyek segítségével tetszőleges mintermek tetszőleges VAGY kapcsolata előállítható (DNF alakot), ezáltal bármilyen kombinációs logikai hálózat realizálható volt (természetesen adott bemenet, ill. kimenet szám mellett).
- A programozható ÉS / VAGY tömbökben úgynevezett „programozható kapcsolók” vannak elhelyezve a horizontális/ vertikális vonalak metszéspontjában.
- Amennyiben a Q_n kimenet(ek)re tárolókat kötünk (pl. egyszerű D tárolót), majd pedig visszacsatoljuk a programozható logikai hálózat bemenete(i)re akár egy sorrendi hálózati viselkedést is meghatározhatunk.

Programozásuk (Fuse) biztosítékok segítségével

- Az összeköttetés mátrix metszéspontjaiban akár kis biztosítékok (fuse) helyezkednek el. Gyárilag pl. logikai '1'-est definiál, tehát vezetőképes. Ha valamilyen spec. programozó eszközzel, a küszöbnél nagyobb feszültséget kapcsolunk rá, átégethető, tehát szigetelővé (nem-vezető) válik, és logikai '0'-át fog reprezentálni.
- A biztosíték átégetése, csak egyszer lehetséges, utána már csak a programozott állapotot fogja tárolni (**OTP** – One time programmable IC).

Példa: PLA tervezése

- Realizálja a következő függvényeket:

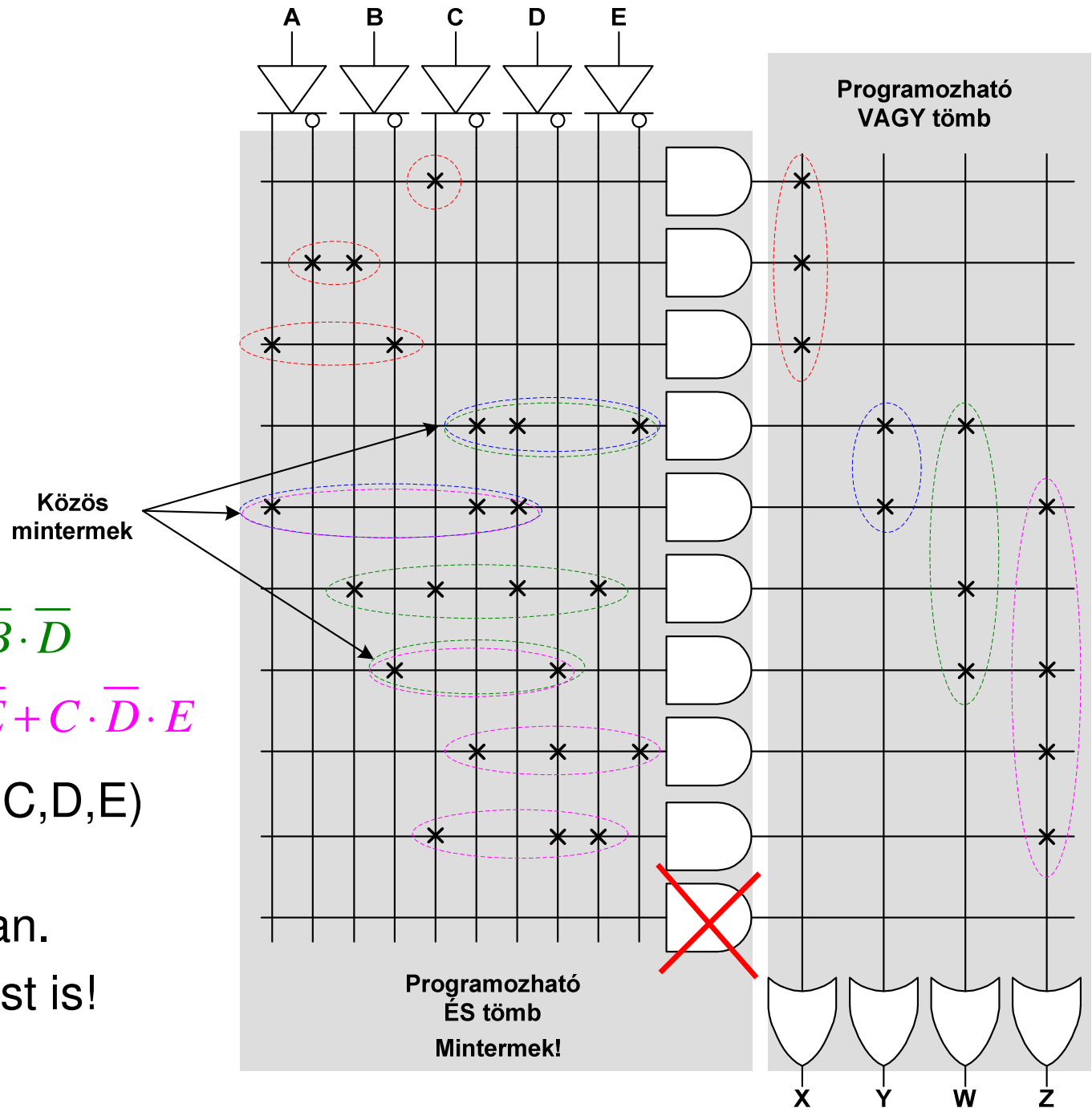
$$X = C + \bar{A} \cdot B + A \cdot \bar{B}$$

$$Y = \bar{C} \cdot D \cdot \bar{E} + A \cdot \bar{C} \cdot D$$

$$W = \bar{C} \cdot D \cdot \bar{E} + B \cdot C \cdot D \cdot E + \bar{B} \cdot \bar{D}$$

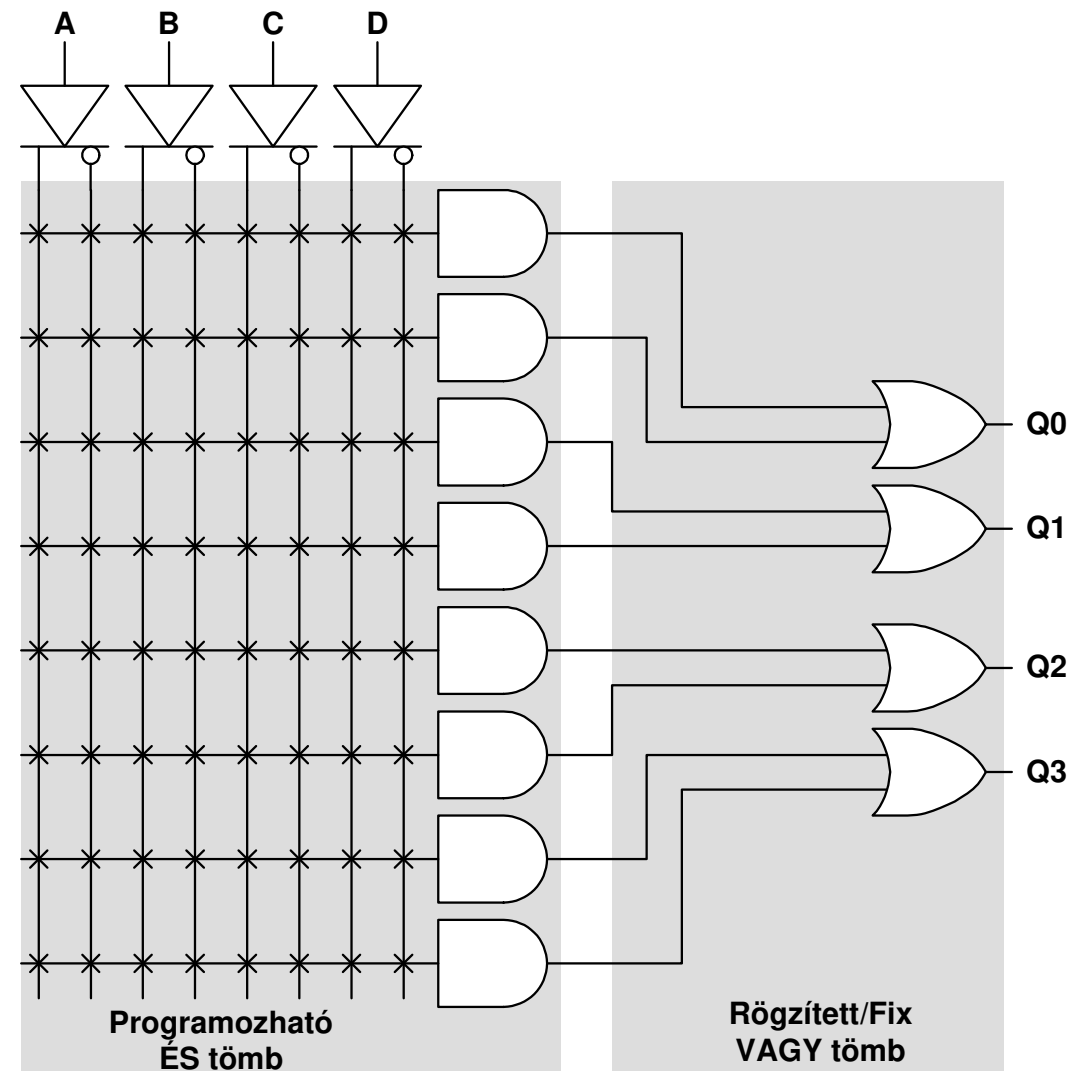
$$Z = A \cdot \bar{C} \cdot D + \bar{B} \cdot \bar{D} + \bar{C} \cdot \bar{D} \cdot \bar{E} + C \cdot \bar{D} \cdot E$$

- Tehát 5 bemenete (A,B,C,D,E) és
- 4 kimenete (X,Y,W,Z) van.
- Rajzoljuk fel a kapcsolást is!



Programmable AND Logic (PAL)

- Egy programozható rész - AND / míg az OR fix
- Véges kombinációja áll elő az AND / OR kapcsolatoknak
- Metszéspontokban kevesebb kapcsoló szükséges
- Gyorsabb, mint a PLA
- Q_n kimeneteken D tárolók (visszacsatolódhatnak a bemenetekre)



PAL

- Elsőként, **1978-ban** az *MMI* (Monolithic Memories Inc.) jelent meg ilyen programozható eszközökkel, majd pedig későbbi jogutódja a *Lattice Semiconductor*, illetve az *AMD* a 80'-as évek végén.
- A PAL hálózatban a programozható részt az ÉS tömb jelenti, míg az VAGY tömb fix/rögzített. Így a tetszőleges mintermeknek csak egy véges kombinációja (VAGY) állítható elő: a lehetséges kimeneti függvények variálhatóságából veszítünk, cserébe viszont a VAGY részek dedikált útvonalainak jelterjedési sebessége nagyobb, míg az eszköz mérete kisebb és ezáltal olcsóbb is lesz.
- Ezáltal a metszéspontokban kevesebb kapcsoló szükséges („gyorsabb”, mint a PLA). Hasonlóan a PLA-khoz, amennyiben a Q_n kimenet(ek)re tárolókat kötünk (pl. egyszerű D tárolót), majd pedig visszacsatoljuk a programozható PAL logikai hálózat bemenete(i)re akár sorrendi hálózati viselkedést is könnyen valósíthatunk.

GAL (Generic Array Logic): Általános tömb logika

- **1985-ben** a *Lattice Semiconductor* fejlesztette ki elsőként,
 - amely a *PAL*-nak egy továbbfejlesztett változatát képviseli.
 - Ugyanolyan belső struktúrával rendelkezik, mint egy *PAL* áramkör,
 - azonban *többször programozható*: tehát törölhető és újraprogramozható eszköz.
 - EEPROM technológiát (lebegő-gate) alkalmaz. Később a National Semiconductor, és AMD is megjelent saját *GAL* sorozataival a piacon.

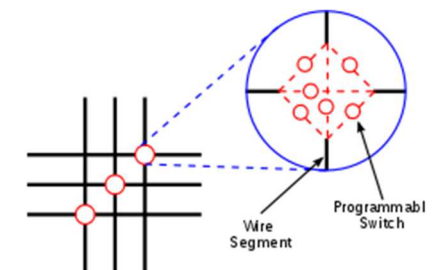
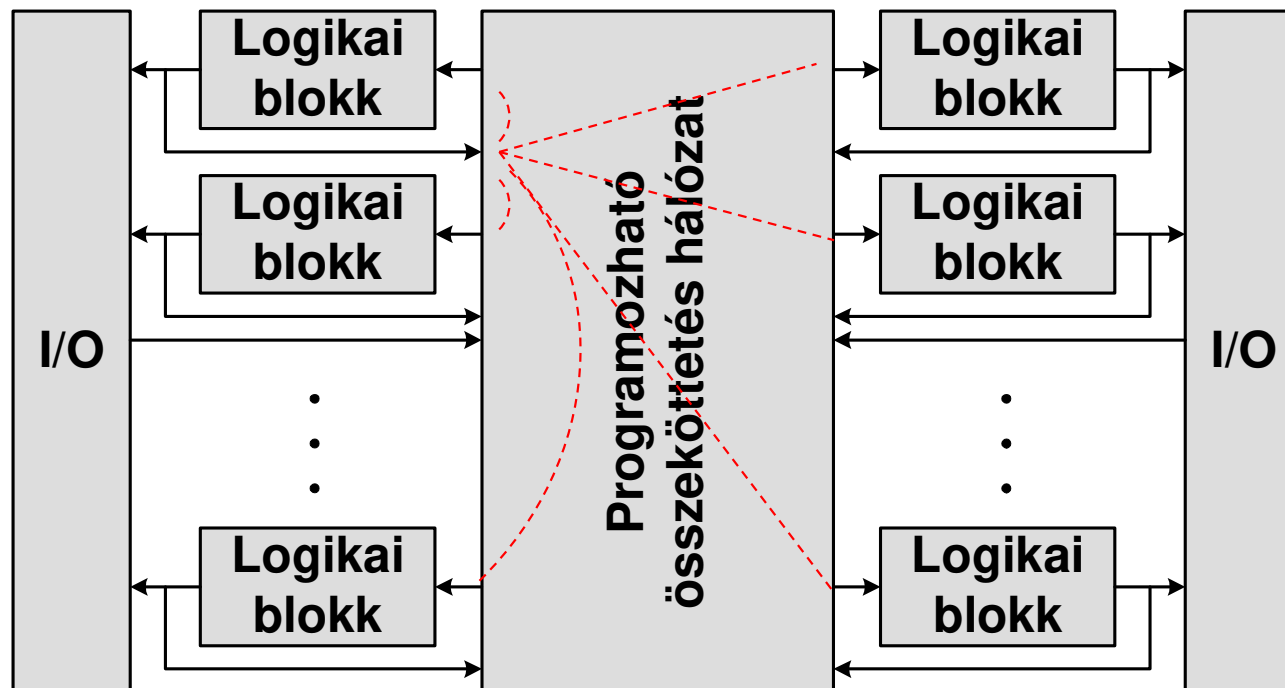
CPLD

CPLD (Complex Programmable Logic Devices): Komplex-programozható Logikai eszközök

- Valójában *átmenetet* képeznek a kis/közepes integráltsági fokú makrocellás PLD-k GAL/PAL áramkörei, illetve a nagy integráltsági fokú FPGA kapu-áramkörök között.
- A GAL/PAL áramköröktől architektúráisan annyiban különbözik, hogy ki lett bővítve: nem egy-, hanem több logikai cellamátrixot tartalmaz, amelyek konfigurálható blokkok reguláris struktúrájában vannak elrendezve. A mai modern FPGA áramköröktől viszont az különbözteti meg felépítésben, hogy *nem tartalmaz dedikált* erőforrásokat (pl. szorzók, memória blokkok).
- a legnagyobb gyártók, amelyek jelenleg is aktív szereplői a CPLD-k piacának a következők: *Xilinx, Intel-Altera, Lattice Semiconductor, MicroChip-Atmel* stb.

Complex Programmable Logic Device (CPLD)

- 1 Logikai Blokkon belül:
 - ~ PAL / PLA (K.H.)
 - Regiszterek (D-FF)
- I/O Blokkok
- Programozható összeköttetések (PI: Programmable Interconnection)
 - Teljes (Full-crossbar), vagy
 - Részleges összeköttetés hálózat



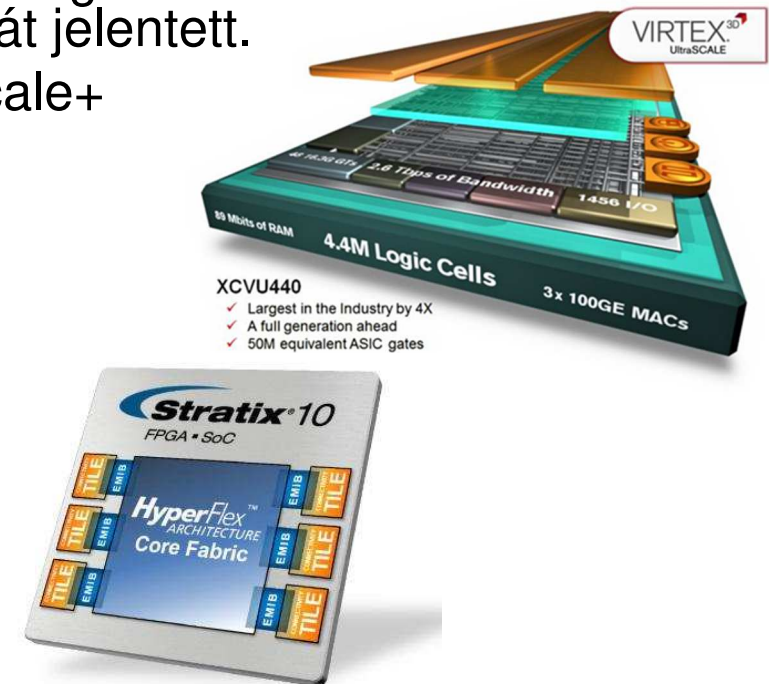
CPLD (folyt)

A CPLD-kben található Logikai Blokk-ok (=makrocellák):

- egyrészt *logikai kapuk* tömbjeit tartalmazzák (hasonlóan a PAL/GAL áramkörök felépítéséhez – DNF alak),
 - másrészt *regisztereket* (D-tárolókból) tartalmaznak a logikai tömbök által előállított kimenetek átmeneti tárolásához, valamint
 - *multiplexereket*, mellyel a programozható összeköttetés hálózatra, vagy I/O blokkok celláihoz lehet továbbítani a belső Logikai Blokkok által előállított kimeneti értékeket. Ezáltal nemcsak logikai kombinációs hálózatokat, hanem akár sorrendi hálózatokat is egyszerűen megvalósíthatunk CPLD-k segítségével
-
- A CPLD-kben található *Programozható összeköttetés hálózat*
 - teljes összeköttetést (mindenki-mindekivel), vagy
 - részleges összeköttetést (valamilyen struktúra szerint, pl. bemenetet – kimenettel, főként régi CPLD típusok esetén) biztosít az egyes blokkok között.
 - Kikapcsoláskor a CPLD konfigurációs memóriája megtartja értékét (non-volatile típus), ezért nem kell egy külső pl. ROM memóriát használni az inicializációs minták tárolásához, bekapcsoláskor ezek automatikusan betöltésre kerülnek. A CPLD-eket közkedvelten alkalmazzák különböző interfészek jeleinek összekapcsolásához (*glue-logic*), amennyiben a jeleken átalakításra is szükség van, továbbá áraik az FPGA-k árainál jóval kedvezőbbek

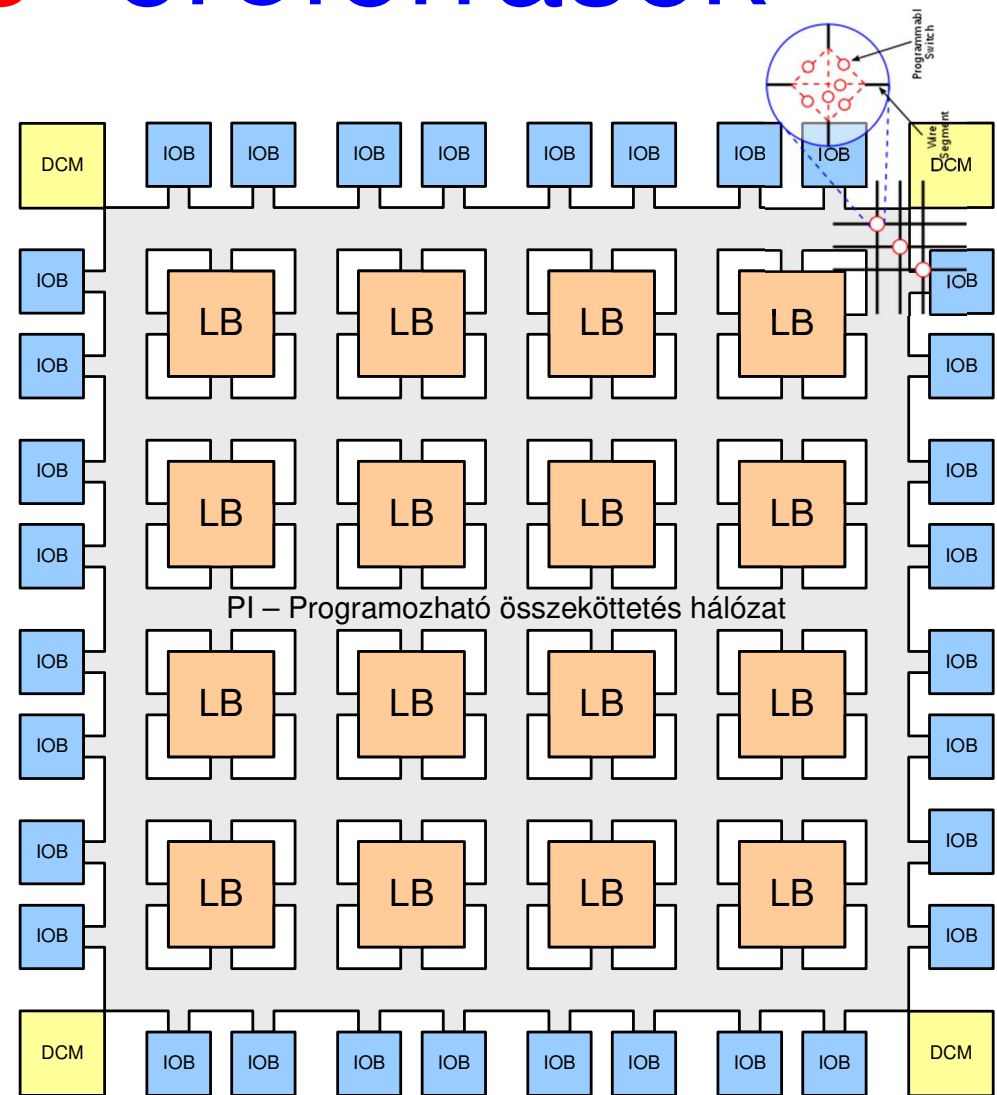
FPGA

- **Field Programmable Gate Array** = „Felhasználó által tetszőlegesen/többször” programozható kapuáramkörök
 - architektúráisan tükrözik mind a PAL, ill. CPLD felépítését, komplexitásban pedig a CPLD-eket is felülmúlják. Nagy, illetve nagyon-nagy integráltsági fokkal rendelkeznek: ~10.000 - ~10.000.000 !! *ekvivalens logikai kaput* is tartalmazhat gyártótól, és sorozattól függően.
 - *Ekvivalens tranzisztorszám*
 - Xilinx Virtex-7 2000T FPGA esetén már meghaladta a ~6.5 milliárdot (2012 – 28nm), amely ~2 millió logikai cellát jelentett.
 - a kapható legnagyobb Xilinx Virtex-Ultrascale+ XCVU440 (2015 – 20nm->16nm) FPGA:
~**20 milliárd tr.** - ~**4.4** millió logikai cella!
(~ 50 millió logikai kapu ekv.)
 - Intel/Altera Stratix-10 **30 milliárd tranz.**
(2016, ~**5.5** millió logikai cella,
4mag ARM-Cortex A53, 14nm)
 - Xilinx Virtex VU19P, **35 milliárd tranz.**
(2019, ~**9** millió logikai cella)



FPGA: „általános” erőforrások

- **LB/CLB:** Konfigurálható Logikai Blokkok, amelyekben LUT-ok (Look-up-table) segítségével realizálhatók például tetszőleges, több bemenetű (ált. 4 vagy 6), egy-kimenetű logikai függvények. Ezek a kimeneti értékek szükség esetén egy-egy D flip-flopban tárolhatók el; továbbá multiplexereket, egyszerű logikai kapukat, és összeköttetéseket is tartalmaznak.
- **IOB:** I/O Blokkok, amelyek a belső programozható logika és a külvilág között teremtenek kapcsolatot. Programozható I/O blokkok kb. 30 ipari szabványt támogatnak (pl. LVDS, LVCMOS, LVTTL, SSTL stb.).
- **PI:** az FPGA belső komponensei között a programozható összeköttetés hálózat teremt kapcsolatot (lokális, globális és regionális útvonalak segítségével, melyeket konfigurálható kapcsolók állítanak be)
- **DCM:** Digitális órajel menedzselő áramkör, amely képes a külső bejövő órajelből tetszőleges fázisú és frekvenciájú belső órajel(ek) előállítására



FPGA – „dedikált” erőforrások

Dedikált erőforrások a következők (amelyek az FPGA típusoktól és komplexitásuktól függően nagy mértékben változhatnak):

- **BRAM:** egy/két portos Blokk-RAM memóriák, melyek nagy mennyiségű ($\sim \times 100\text{Kbyte}$ – akár $\sim \times 10\text{Mbyte}$) adat/utasítás tárolását teszik lehetővé, egyenként 18K / 36 Kbites kapacitással *
- **MULT / vagy DSP Blokkok:** beágyazott szorzó áramköröket jelentenek, amelyek segítségével hagyományos szorzási műveletet, vagy a DSP blokk esetén akár bonyolultabb DSP MAC (szorzás-akkumulálás), valamint aritmetikai (kivonás) és logikai műveleteket is végrehajthatunk nagy sebességgel.
- **Beágyazott/Beágyazható processzor(ok)**:**
 - Tetszés szerint konfigurálható / beágyazható ún. *soft-processzor* mag(ok)
 - Példa: Xilinx PicoBlaze, **Xilinx MicroBlaze**, Altera Nios II stb.
 - Fixen beágyazott, ún. *hard-processzor* mag(ok)
 - Példa: IBM PowerPC 405/450 (Xilinx Virtex 2 Pro, Virtex-4 FXT, Virtex-5 FXT), **ARM Cortex A9 (Xilinx Zynq, illetve Altera Cyclone V, Stratix V, Arria V, MicroSemi Smartfusion-1,-2 FPGA chipjei),**) stb.

* FPGA függő adatok (Xilinx)

** 2015-ös adatok szerint

Xilinx FPGA családok *

■ Nagy teljesítmény

- Virtex (1998)
 - 50K-1M kapu, 0.22μm
- Virtex-E/EM (1999)
 - 50K-4M kapu, 0.18μm
- Virtex-II (1999)
 - 40K-8M kapu, 0.15μm
- Virtex-II Pro/X (2002)
 - 50K-10M kapu, 0.13μm
- Virtex-4 (2004) [LX, FX, SX]
 - 50K-10M kapu, 90nm
- Virtex-5 (2006) [LX, LXT, SXT]
 - 65nm
- Virtex-5 FXT, TXT (2008)
 - 65nm
- Virtex-6 LXT, SXT (2009)
 - 40nm
- Virtex-7 (2011)
 - 28nm

□ Kintex-7 (2011)
■ 28nm

■ Alacsony költség

- Spartan-II (2000)
 - 15K-200K kapu, 0.22μm
- Spartan-IIE (2001)
 - 50K-600K kapu, 0.18μm
- Spartan-3 (2003)
 - 50K-5M kapu, 90nm
- **Spartan-3E (2005)**
 - **100K-1.6M kapu, 90nm**
- Spartan-3AN (2006)
 - 50K-1.4M kapu, 90nm
- Spartan-3A - DSP (2006)
 - 1.8M-3.4M kapu, 90nm
- Spartan-6 LX, LXT (2009)
 - 45nm
- ...
- Artix-7 (2011)
 - 28nm
- **Zynq-7000 APSoC (2012)**
 - **28nm**
- UltraScale MPSoC (2015)
 - 16nm
- Spartan-7 (2017)
 - 28nm

Pannon Egyetem - VIRT

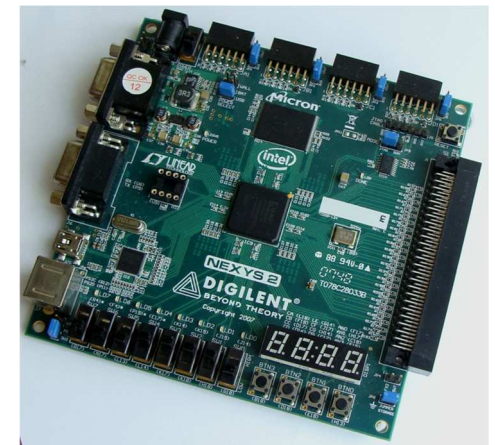
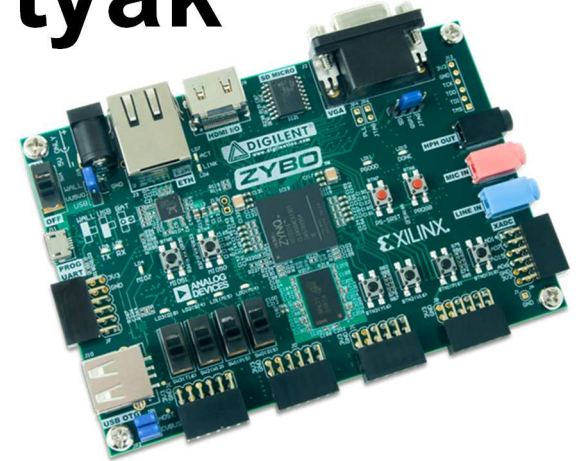
Tanszéken lévő fejlesztő kártyák

HW:

- ***Digilent ZyBo (Xilinx Zynq)***
- ***Digilent Nexys-2 (Xilinx Spartan-3E)***
- Digilent Atlys (Xilinx Spartan-6)
- Digilent Nexys-4 (Xilinx Artix-7)
- Xilinx Virtex-5 ML506
- Virtex2-Pro (Xilinx Virtex2-Pro)
- RC203/RC200 (Xilinx Virtex-II)

SW:

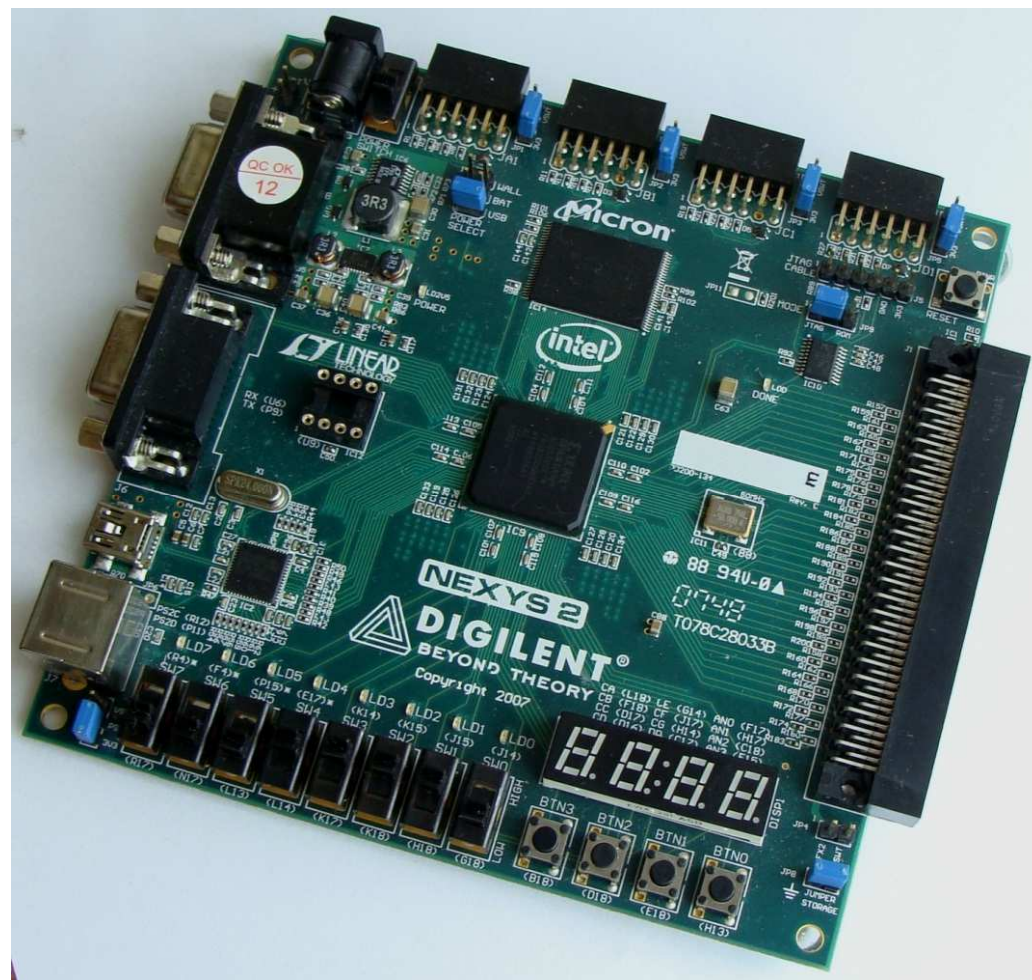
- Xilinx ISE Design Suite 14.7
- Xilinx Vivado HLx 2018.3



Digilent Nexys-2 fejlesztő kártya

Nexys™2 Xilinx Spartan-3E FPGA fejlesztő kártya

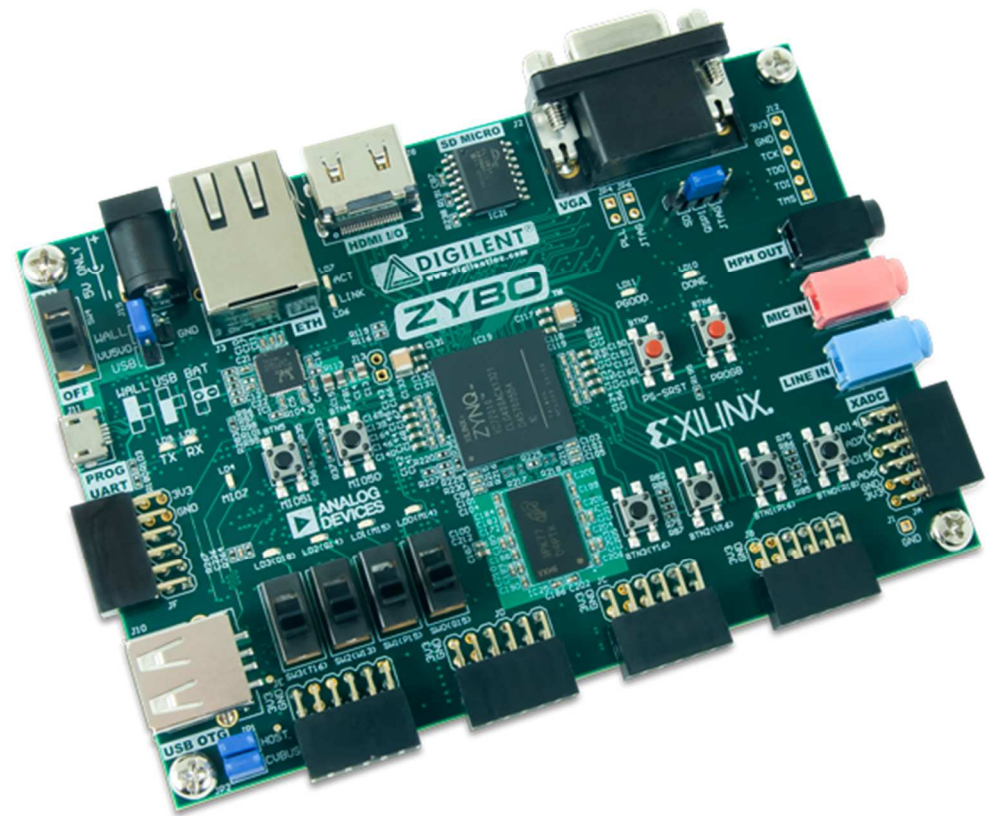
- Xilinx Spartan-3E FPGA, 500K / 1200K ekvivalens kapuval
- USB2 port (táp, konfiguráció, adat-transzfer egyben)
- Xilinx ISE/Webpack/EDK
- 16MB Micron PSDRAM
- 16MB Intel StrataFlash Flash
- Xilinx Platform Flash ROM
- 50MHz osszcillátor
- 75 FPGA I/O's (1 nagy-sebességű Hirose FX2 konnektor és 4 db 2x6 PMOD konnektor)
- GPIO: 8 LED, 4-jegyű 7-szegmenses kijelző, 4 nyomógomb, 8 kapcsoló
- VGA, PS/2, Soros port



ZYBO™ Zynq FPGA fejlesztő kártya

- **Xilinx Zynq-7000 (Z-7010)**
 - 650 MHz dual ARM Cortex-A9 magok (PS)
 - 8-csatornás DMA vezérlő (PS)
 - 1G ethernet, I2C, SPI, USB-OTG vezérlő (PS)
 - **Artix-7** FPGA logika (PL), 54 muxed MIO pins (PL/PS)
 - 28Kbyte logikai cella, 240 Kbyte BRAM, 80 DSP szorzó(PL)
 - 12-bites, 1MSPS XADC (PL)
- 512 MByte DDR3 x32-bit (adatbusz), 1050Mbps sávszélességgel
- Tri-mode 10/100/1000 Ethernet PHY
- HDMI port: Dual role (source/sink)
- VGA port: 16-bites
- uSD kártya: OS tartalom tárolása
- OTG USB 2.0 (host és device)
- Audio codec
- 128Mbit x Serial Flash/QSPI (konfiguráció tárolási célokra)
- JTAG-USB programozhatóság, UART-USB vezérlő
- GPIO: 5 LED, 6 nyomógomb, 4 kapcsoló
- 4+1 PMOD csatlakozó (A/D átalakítóhoz)

Web: <http://digilentinc.com/Products/Detail.cfm?NavPath=2,400,1198&Prod=ZYBO>





„FPGA” témájú tárgyak a Pannon Egyetemen – VIRT tanszék

Mérnök informatikus BSc-MSc / Villamosmérnök BSc képzés

- **(VEMIKN5143D) Tervezési módszerek programozható logikai eszközökkel - VHDL (őszii félév)**
- **(VEMIVI4144B) FPGA-alapú Beágyazott rendszerek (tavaszi félév)**

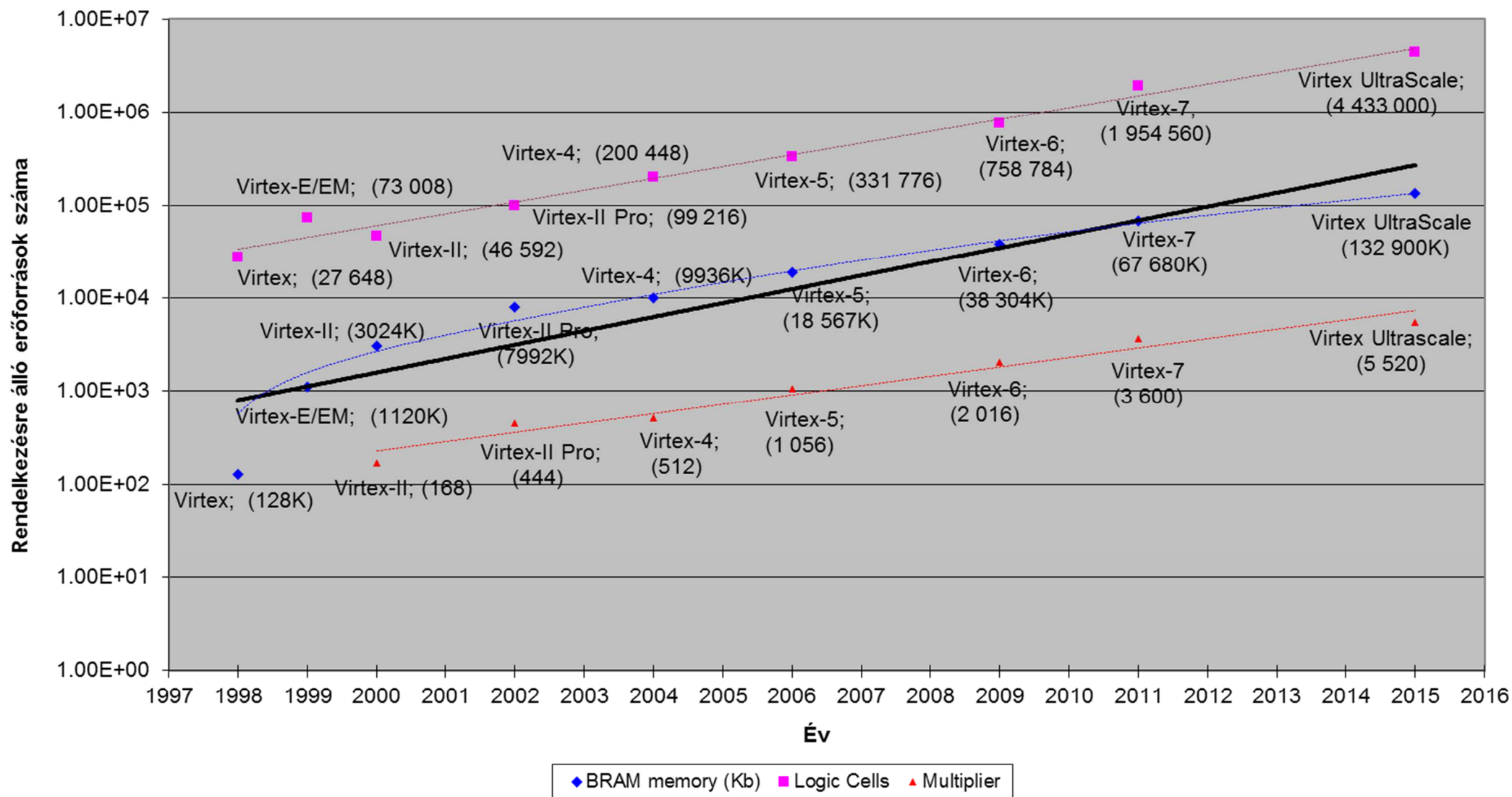
FPGA létjogosultsága?

A mai modern **FPGA-k** a

- nagyfokú flexibilitásukkal,
 - nagy számítási teljesítményükkel,
 - és (ASIC = Alkalmazás Specifikus Áramkörökkel szemben) gyors prototípus-fejlesztési,
 - ezáltal olcsó kihozatali (piacra kerülési) költségükkel
- igen jó alternatívát teremtenek a *mikrovezérlős* (uC), illetve *DSP-alapú* implementációk leváltására (pl. jelfeldolgozás, hálózati titkosítás, beágyazott rendszerek, stb. alkalmazásai területén).

Fejlődésüket jól tükrözi a mikroprocesszorok és az FPGA/PLD áramköri technológia fejlődési üteme között fennálló hasonlóság a méretcsökkenésnek („**scaling-down**”) - *Gordon Moore-törvénynek* megfelelően (1965!).

Nagy-teljesítményű Xilinx Virtex FPGA család erőforrásainak alakulása (1998-2015 időszakban)

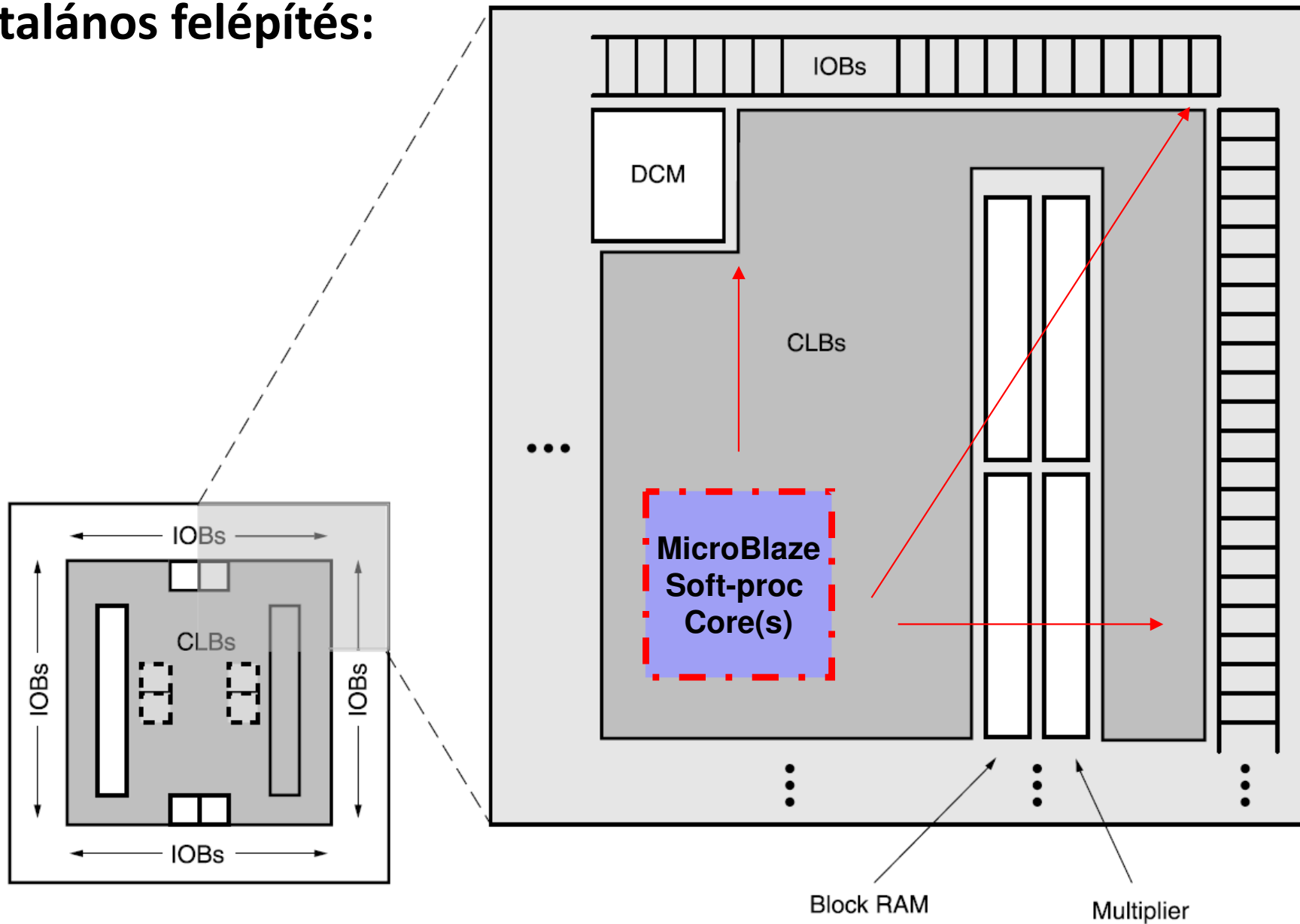


*Tranzisztorok fizikai méretcsökkenésének („Scaling-down”) hatása a fejlődésre



Pl. Spartan-3E architektúra

Általános felépítés:



Spartan-3E FPGA építőelemei

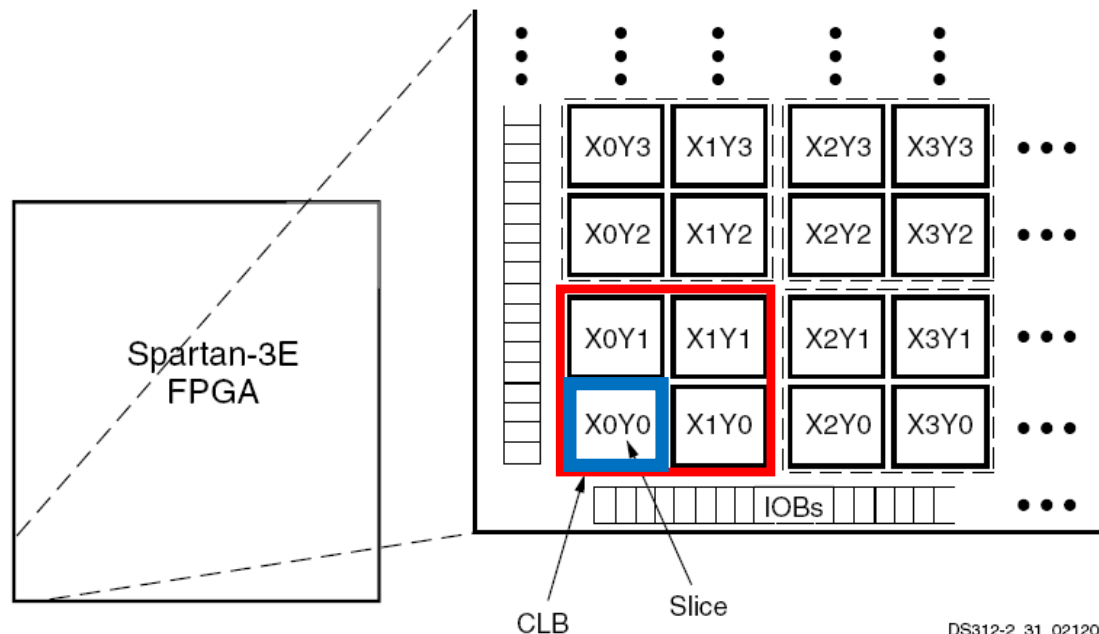


- **CLB:** Konfigurálható Logikai blokkok
 - Slice-L, mint *Logika*: LUT, D-FF, MUX, Carry Logika
 - Slice-M, mint *Memória*: SRL-16x1, RAM-16x1
- **IOB:** I/O blokkok
- **MULT:** 18x18 bites előjeles (2's) szorzó(k)
- **BRAM:** konfigurálható 18 Kbytes (~2 Kbyte + paritás) egy/két-portos memóri(ák)
- **DCM:** Digitális órajel menedzser blokkok
- **Beágyazható processzor(ok):**
 - Csak szoft-processzor mag(ok) alakíthatóak ki
 - Pl: *Xilinx MicroBlaze, PicoBlaze*, (esetleg külső független IP – szellemi terméke integrálható)

Spartan-3E CLB tömb

CLB: Konfigurálható
Logikai Blokkok: fő
logikai erőforrás,
kombinációs és
szekvenciális logikai
hálózatok tervezésére

**1 CLB = 4
Slice !**



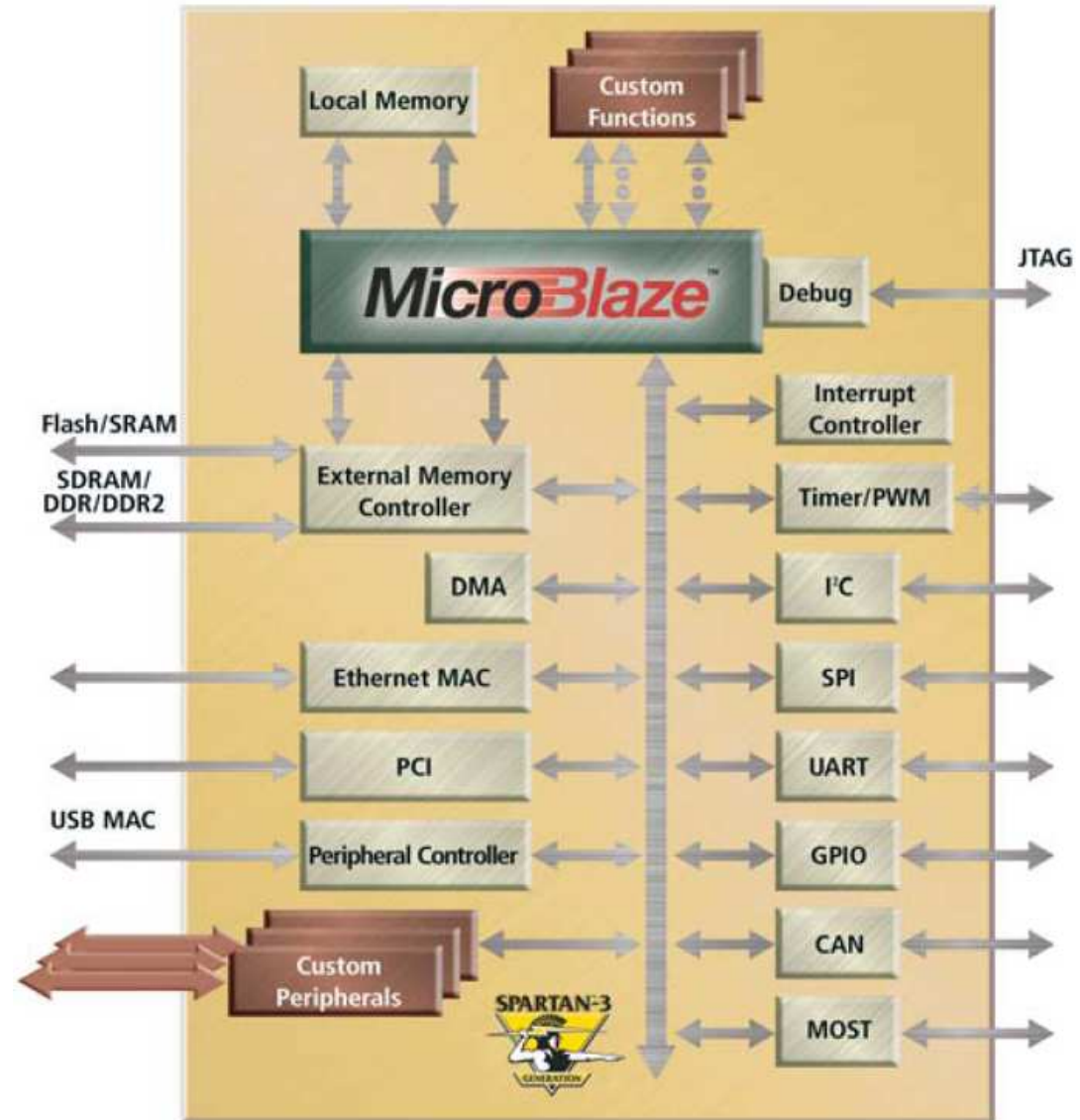
DS312-2_31_021205

Device	CLB Rows	CLB Columns	CLB Total ⁽¹⁾	Slices	LUTs / Flip-Flops	Equivalent Logic Cells	RAM16 / SRL16	Distributed RAM Bits
XC3S100E	22	16	240	960	1,920	2,160	960	15,360
XC3S250E	34	26	612	2,448	4,896	5,508	2,448	39,168
XC3S500E	46	34	1,164	$1,164 \cdot 4 = 4,656$	$4,656 \cdot 2 = 9,312$	10,476	4,656	$4,656 \cdot 16 = 74,496$
XC3S1200E	60	46	2,168	$2,168 \cdot 4 = 8,672$	$8,672 \cdot 2 = 17,344$	19,512	8,672	$8,672 \cdot 16 = 138,752$
XC3S1600E	76	58	3,688	14,752	29,504	33,192	14,752	236,032

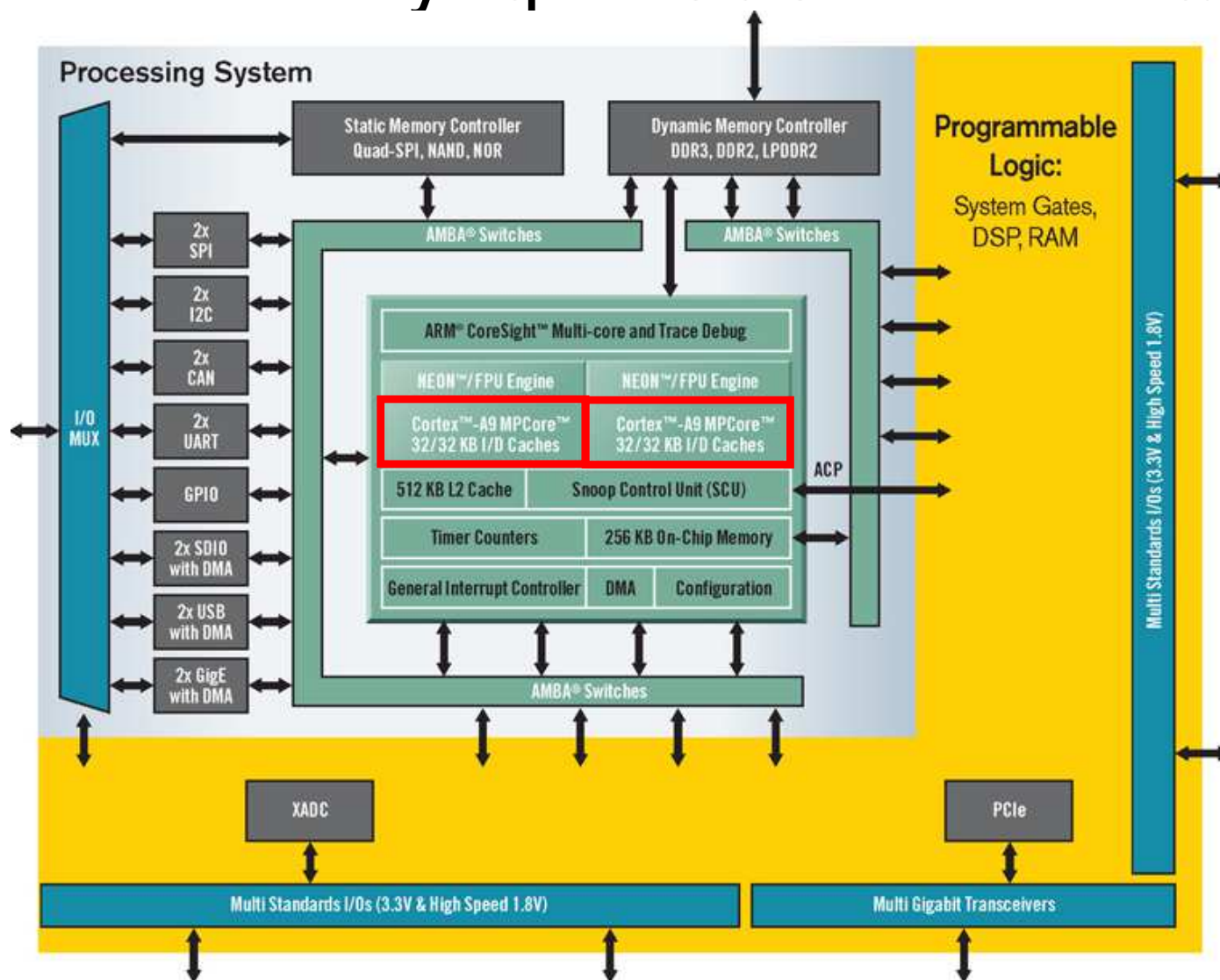
MicroBlaze szoft-processzor mag

„Beágyazható” processzor*

- **RISC utasítás készlet architektúra**
- **32-bites szoft-processzor mag**
- **133+ MHz órajel (PLB busz)**
- **Harvard blokk-architektúra**
- Kis fogyasztás: ~ mW/MHz
- 3/5 lépcsős adatvonal pipe-line
- 32 darab 32-bites általános célú regiszter
- utasítás Cache / adat Cache
- Időzíteni lehetőségek (timer)
- Sokféle periféria, kommunikációs interfész csatlakoztatható (IP core-ok)
- **Minden** Xilinx FPGA-n implementálható, melynek elegendő erőforrása van és a fejlesztő szoftver támogatja!



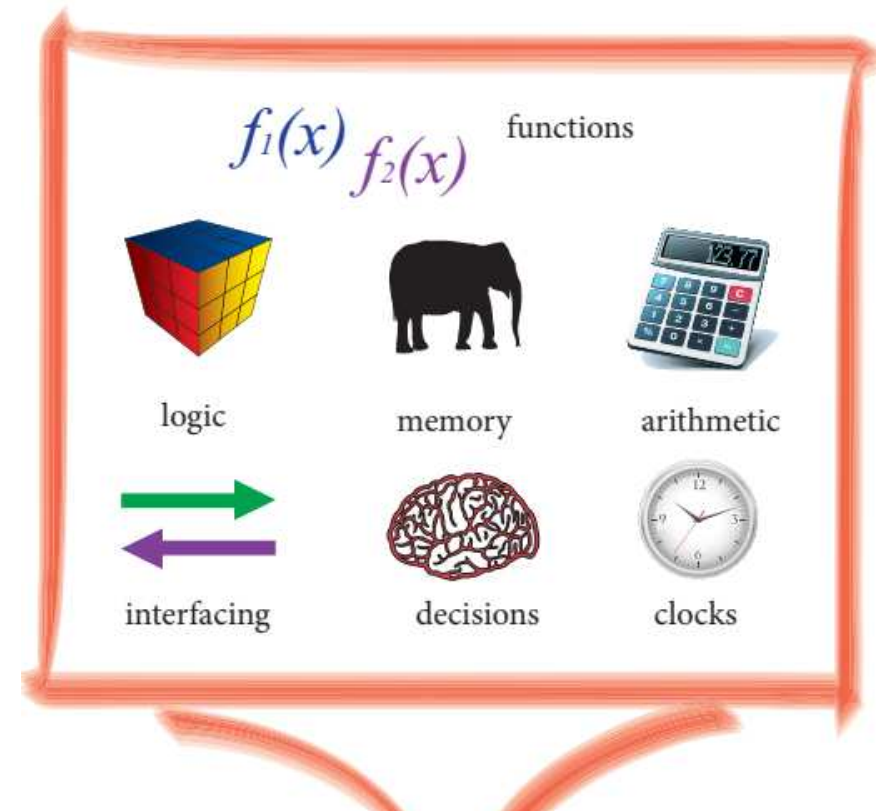
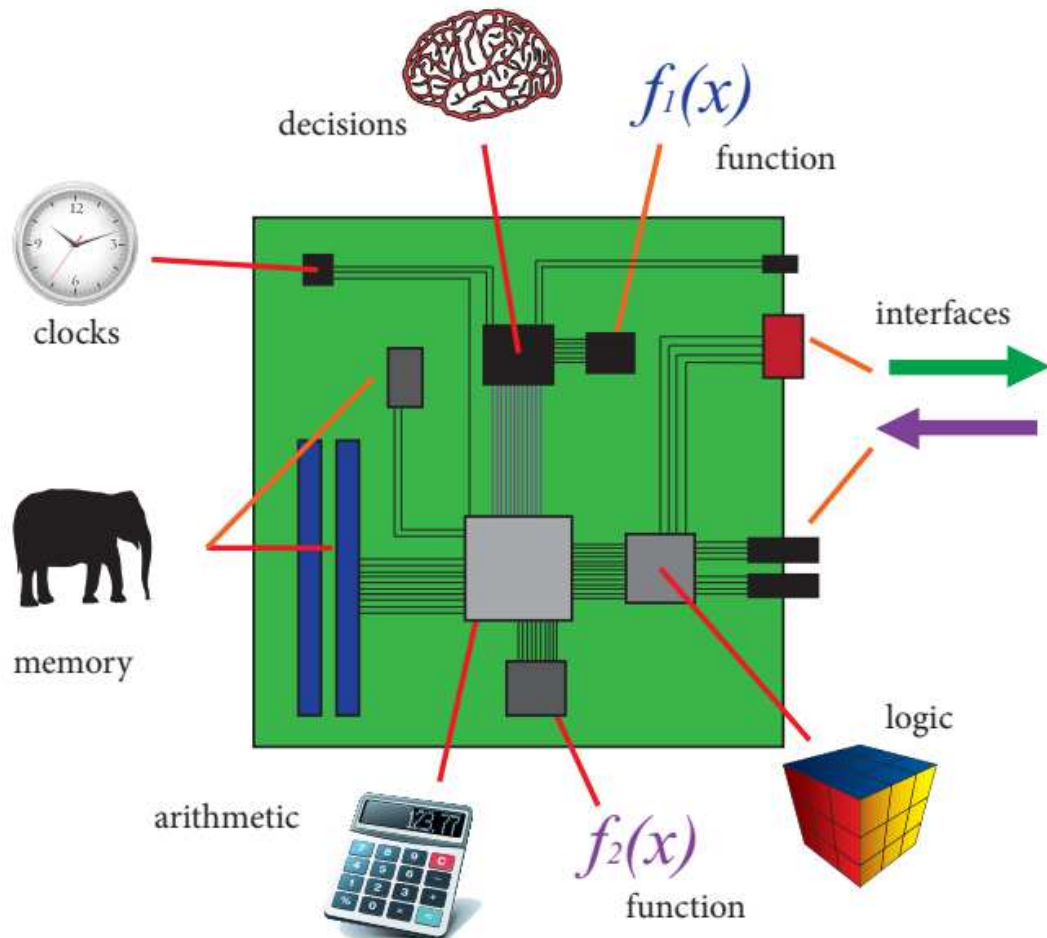
Xilinx Zynq APSoC – ARM hard-procессzor



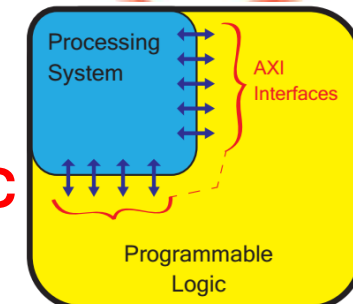
- **Dupla ARM Cortex™-A9 MPCore (650+ MHz)**
- Neon: 32-/ 64-bit FPU
- 32kB utasítás & 32kB adat L1 Cache
- Közös 512kB L2 Cache
- 256kB on-chip memória
- Integrált DDR3, DDR2 and LPDDR2 DDR vezérlő
- Integrált 2x QSPI, NAND Flash and NOR Flash memória vezérlő
- Perifériák: 2x USB2.0 (OTG), 2x GbE, 2x CAN2.0B 2x SD/SDIO, 2x UART, 2x SPI, 2x I2C, 4x 32b GPIO, PCI Express® Gen2 x8
- Két 12-bit 1Msps ADC
- **28nm** Programozható FPGA Logika:
- **28k** - 350k Logikai cella (~ 430k to 5.2M ekvivalens kapu)
- 240KB - 2180KB Block RAM
- 80 - 900 18x25 DSP szorzó (58 - 1080 GMACS -DSP teljesítmény)

APSoC = All Programmable System-On-a-Chip

System-On-a-Board vs. System-On-a-Chip



**Xilinx
Zynq
APSoC**

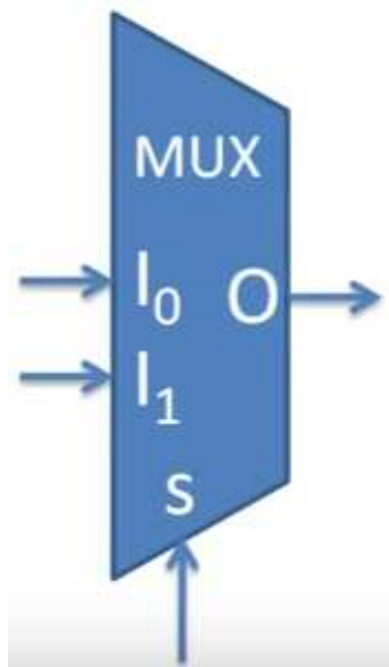




Példák – VHDL szimuláció

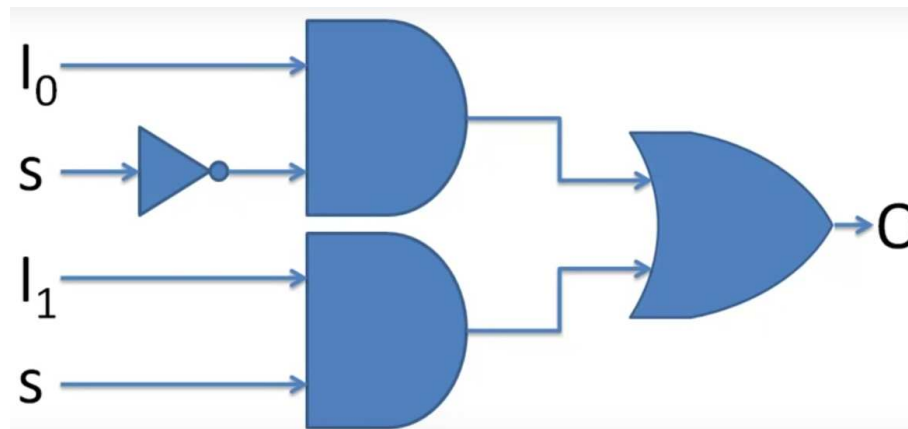
Példa 1.) – VHDL szimuláció

■ 2:1 MUX – statikus hazárd mentesítése



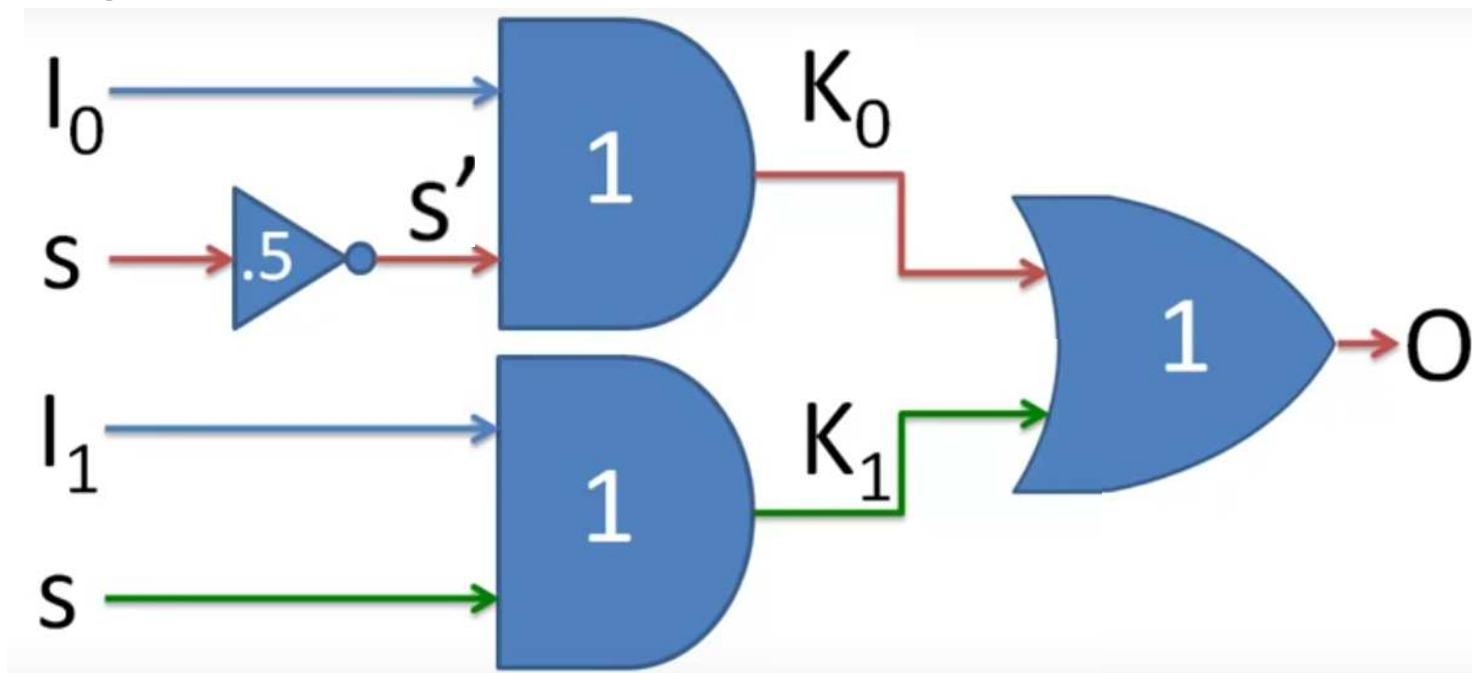
		$I_1 I_0$			
		00	01	11	10
S	0	0	1	1	0
	1	0	0	1	1

$$O = \bar{S} \cdot I_0 + S \cdot I_1$$

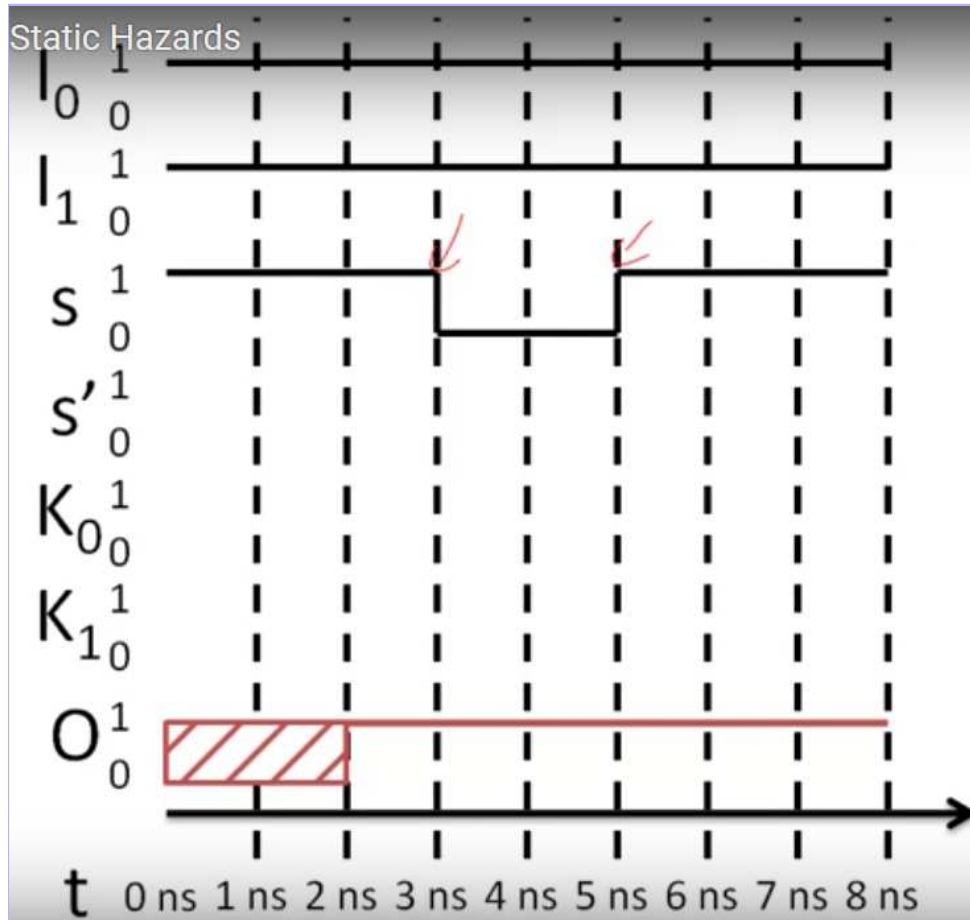
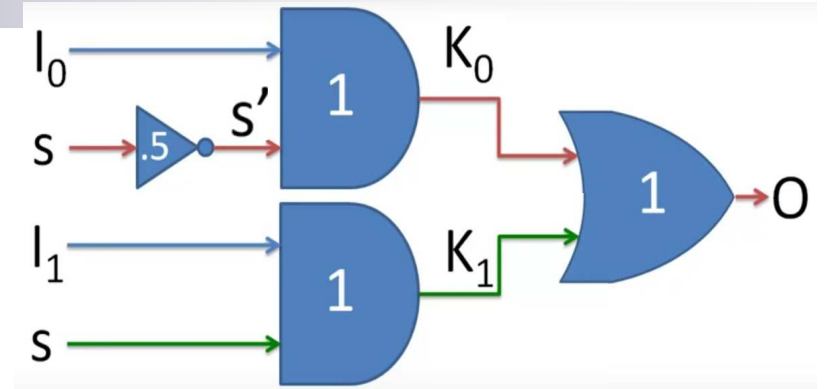


2:1 MUX valós áramkör

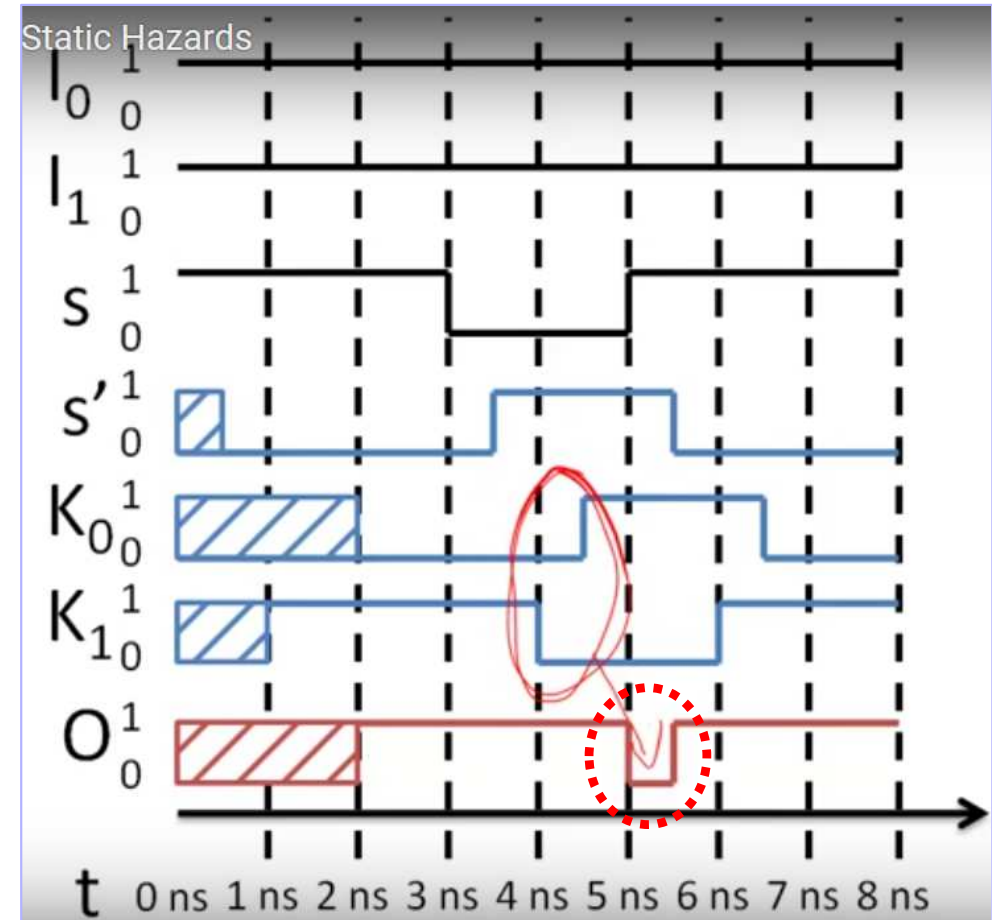
- Egyedi „dt” késleltetések bejelölve a kapukon
 - Legrosszabb útvonal $0.5 + 1 + 1 \text{ ns} = 2.5 \text{ ns}$



Statikus hazárd

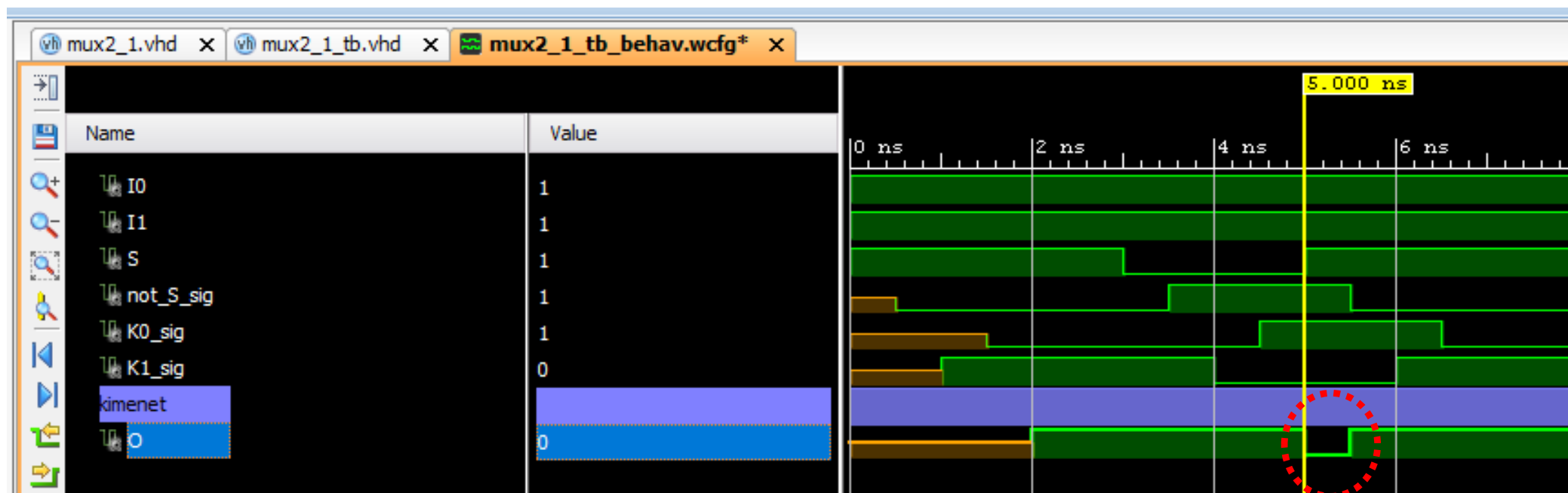


Elvárt kimenet (ideális áramkör)



Számított kimenet (késleltetéssel)

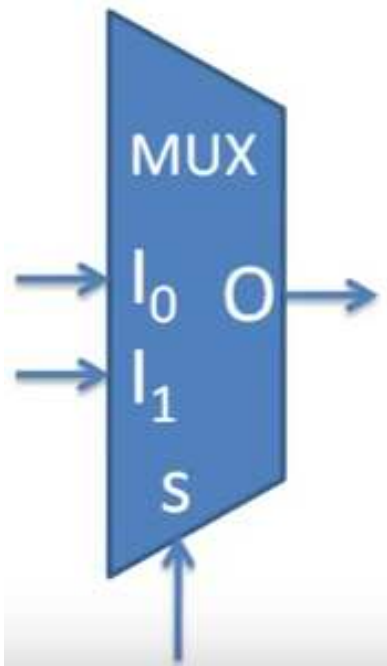
VHDL szimuláció



```
not_S_sig <= transport (not S) after 0.5 ns;  
K0_sig <= transport (I0 and not_S_sig) after 1 ns;  
K1_sig <= transport (I1 and S) after 1 ns;  
----comment out next line to eliminate static hazard  
--K2_sig <= transport (i0 and i1) after 1 ns;  
  
O <= transport (K0_sig or K1_sig) after 1 ns;  
----comment out next line to eliminate static hazard  
--O <= transport (K0_sig or K1_sig or K2_sig) after 1 ns;
```

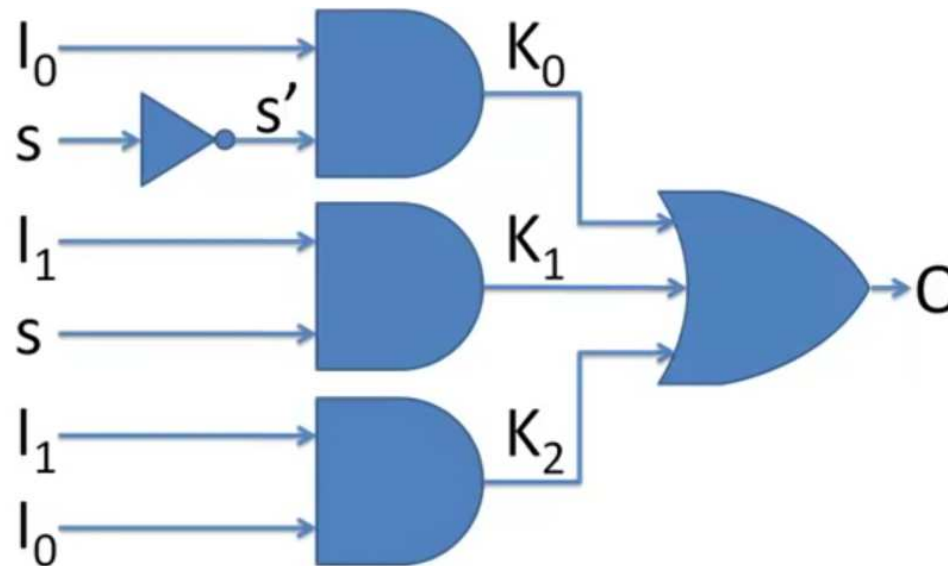
2:1 MUX

■ Statikus hazárd kiküszöbölése

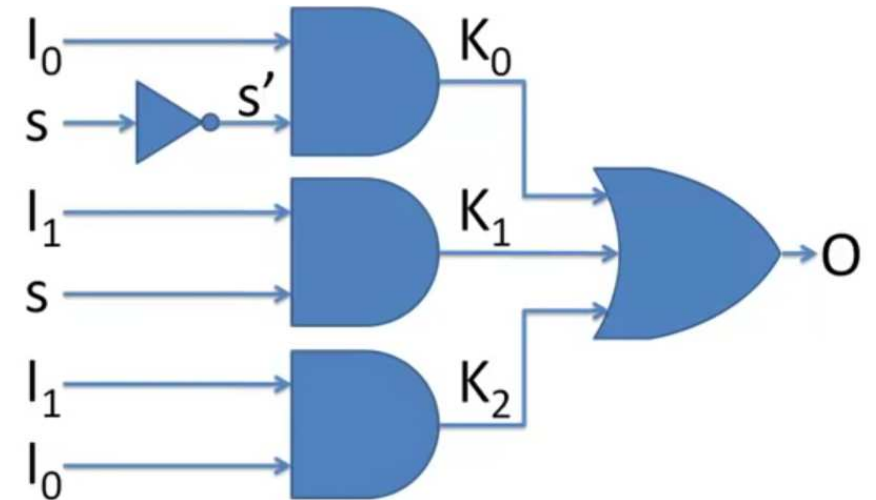
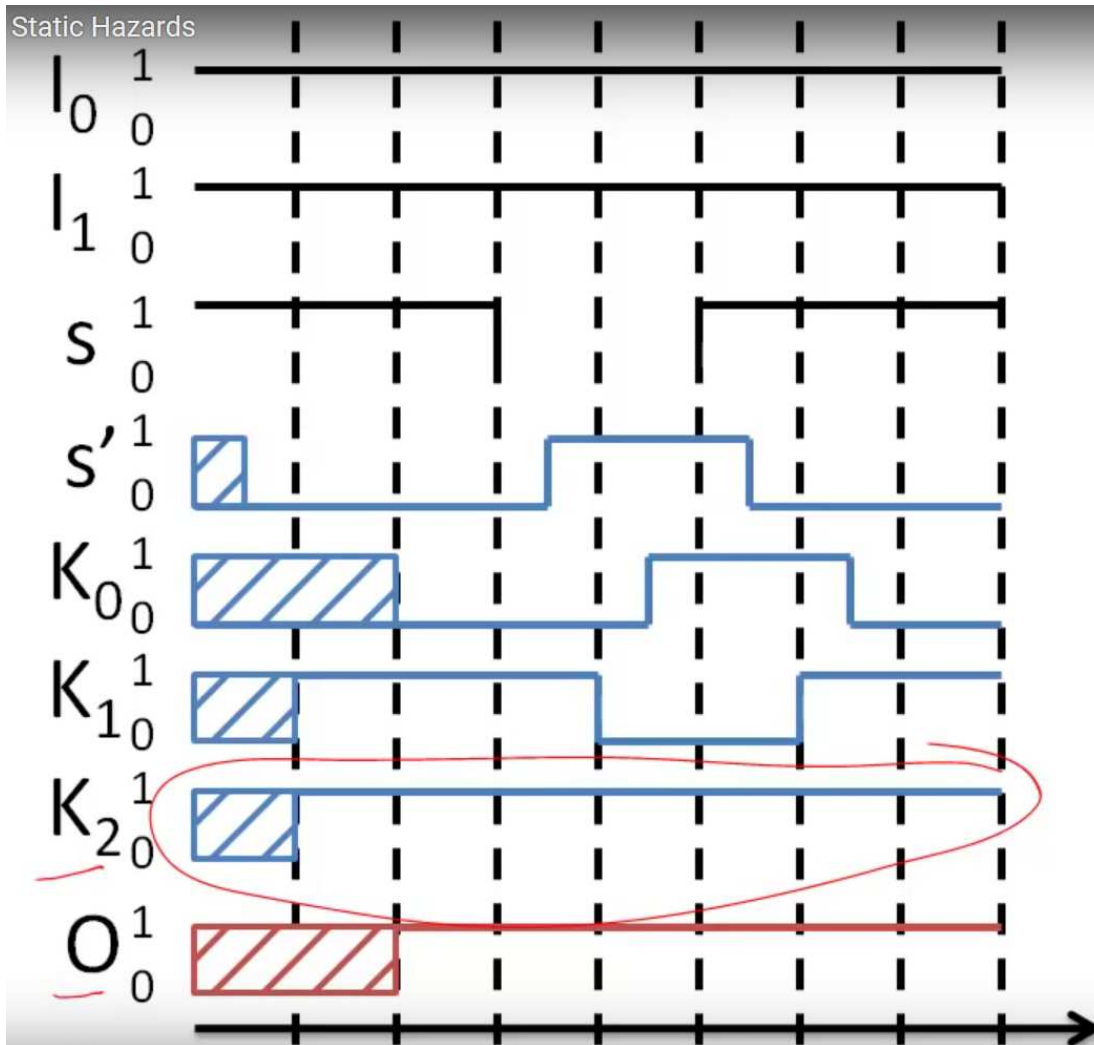


		$I_1 I_0$			
		00	01	11	10
S	0	0	1	1	0
	1	0	0	1	1

$$O = \bar{S} \cdot I_0 + S \cdot I_1 + I_0 \cdot I_1$$

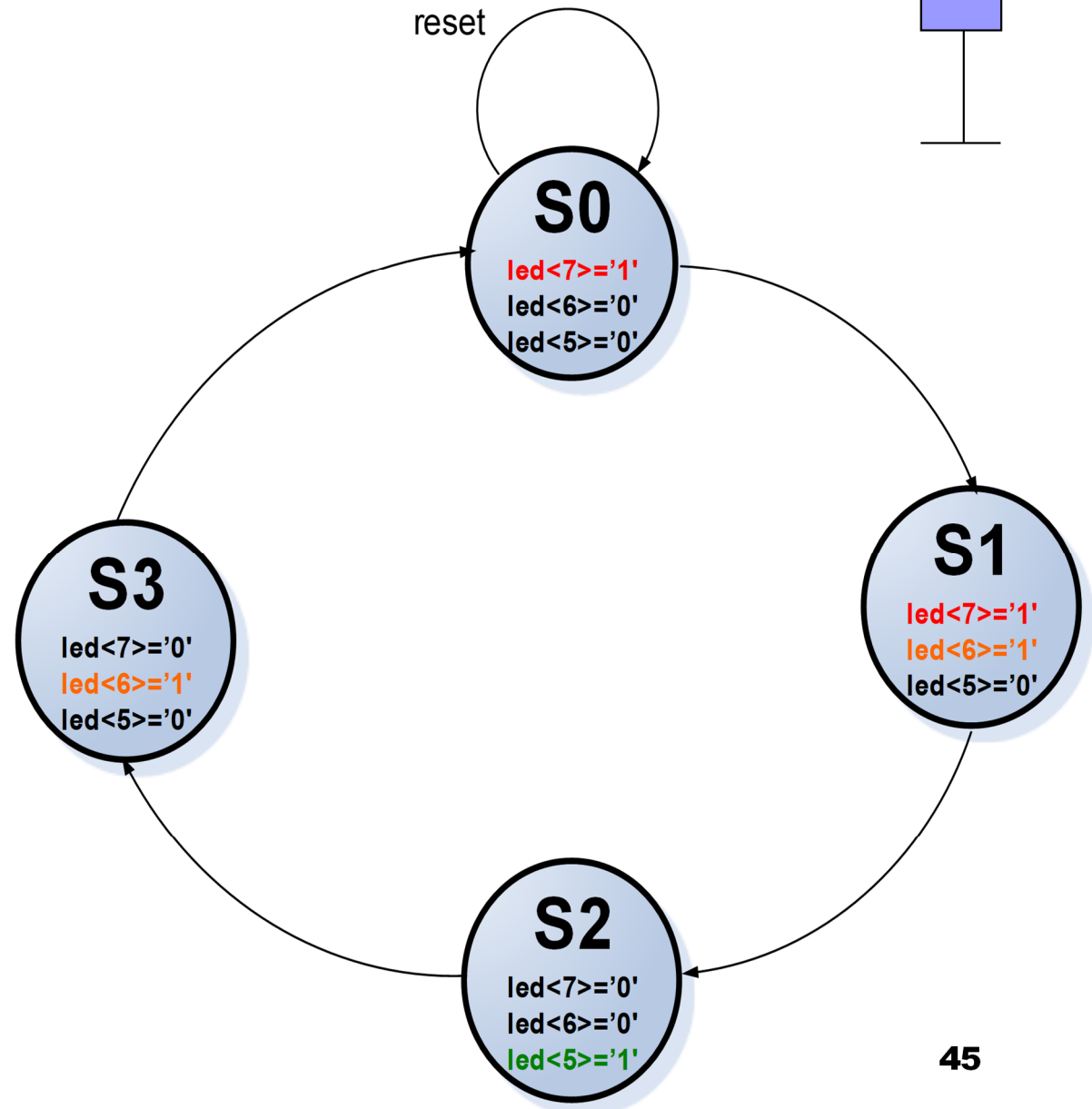


Statikus hazard kiküszöbölése



Példa 2.) – Jelzőlámpa

- Közlekedési lámpa vezérlő = állapotdiagram (FSM – Finite State Machine).
- Megvalósítás: Moore modell-el.



Feladat 1/a.) Megoldás: Moore modell

([traffic_moore.vhd](#))

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

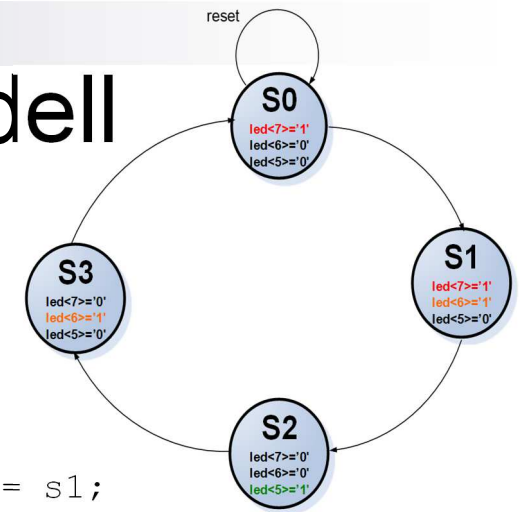
entity traffic_moore is
generic (N: natural := 3);
port ( clk : in  STD_LOGIC;
      reset : in  STD_LOGIC;
      led : out  STD_LOGIC_VECTOR (N-1
        downto 0));
end traffic_moore;

architecture Behavioral of traffic_moore is
type traff_state_type is (s0, s1, s2, s3);
-- P, PS, Z ,S -> P
signal state_reg, state_next:
traff_state_type;

begin
-- state register
process(clk, reset)
begin
if (reset='1') then
state_reg <= s0;
elsif rising_edge(clk) then
state_reg <= state_next;
end if;
end process;
```

```
-- next-state logic
process(state_reg)
begin
case state_reg is
when s0 =>
state_next <= s1;
when s1 =>
state_next <= s2;
when s2 =>
state_next <= s3;
when s3 =>
state_next <= s0;
end case;
end process;
```

```
-- Moore output logic (led2, led1, led0) !
process(state_reg)
begin
case state_reg is
when s0 =>
led(N-1 downto 0) <= "100"; --P
when s1 =>
led(N-1 downto 0) <= "110"; --PS
when s2 =>
led(N-1 downto 0) <= "001"; --Z
when s3 =>
led(N-1 downto 0) <= "010"; --S
end case;
end process;
end Behavioral;
```



Szimulációs eredmény

