Dr. Kincses Zoltán, ***Dr. Vörösházi Zsolt*:**

FPGA-alapú beágyazott rendszerek tervezése

A felsőfokú informatikai oktatás minőségének fejlesztése, modernizációja

TÁMOP-4.1.2.A/1-11/1-2011-0104

Főkedvezményezett:
Pannon Egyetem
8200 Veszprém
Egyetem u. 10.

Kedvezményezett:
Szegedi Tudományegyetem
6720 Szeged
Dugonics tér 13.

2014

A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Frissítve: 2020. február 14.

FPGA-alapú beágyazott rendszerek tervezése

Dr. Vörösházi Zsolt

1. Bevezetés

Oktatási cél

A **Pannon Egyetemen** 2010/11 őszi félévétől az aktuális ipari követelményeknek megfelelő tematikával újraindult a ***„Tervezési módszerek programozható logikai alkatrészekkel (VHDL)”*** c. tárgy Villamosmérnök BSc hallgatóknak, amely egy bevezetést ad az FPGA alapú digitális hálózatok VHDL alapú tervezésébe. A tárgy labor gyakorlatain a hallgatóknak kis csoportokban együtt dolgozva kell a kitűzött feladatokat megoldaniuk és Digilent Nexys-2/Zybo kártyákon implementálniuk, ezáltal is ösztönözve őket a valós elvárásokra: az együttes tervezés, fejlesztés és tesztelés metodikájára.

- Ennek a tárgynak a folytatását képzi a *2011/12* őszi félévétől új tematikával indított ***„FPGA-alapú Beágyazott Rendszerek”*** c. kötelező laboratóriumi tárgy Villamosmérnök BSc hallgatóknak. A jelenlegi fólia gyűjtemény, mind elméleti, mind pedig gyakorlati részeivel – az SZTE*-s képzéssel párhuzamosan – ehhez a tárgyhoz kapcsolódik a jelenlegi tananyag is. Majd *2014/15* őszi félévétől mindkét tárgy elindult a Mérnök Informatikus BSc képzésben is.

Az eddigi oktatási tapasztalatokat és hallgatói visszajelzéseket, valamint az ipari partnerek érdeklődését és igényeit is szem előtt tartva egy olyan hiánypótló előadás vázlat készült, amely nemzetközi szintű alkalmazott szakirodalomra épül. A jegyzet bizonyos részei a *Xilinx Embedded System Design Flow - Professor Workshop and Teaching Materials*, valamint a *Xilinx Embedded Linux on the MicroBlaze Processor* segédanyagaira együttesen épülnek.

Áttekintés

- Bevezetés – Beágyazott rendszerek?
- FPGA (**F**ield **P**rogrammable **G**ate **A**rrays)?
- Használt fejlesztő hardverek és eszközök:
  - **Digilent ZyBo** fejlesztő kártyák
- Használt fejlesztő szoftverek:
  - Xilinx Vivado Design Suite **(**2018.3**)**
    - Embedded + Software Dev. Kit

Tárgyalt ismeretkörök

1. **Bevezetés – Beágyazott rendszerek**
2. FPGA-k, Digilent ZYBO fejlesztő kártyák és eszközök
3. Beágyazott Rendszer fejlesztő szoftverkörnyezet (Xilinx Vivado Embedded Development) áttekintése
4. Beágyazott alap tesztrendszer (BSB - Base System Builder and Board Bring-Up) összeállítása
5. Perifériák hozzáadása (IP adatbázisból) az összeállított beágyazott alaprendszerhez
6. Saját periféria hozzáadása az összeállított beágyazott alaprendszerhez
7. Szoftver alkalmazások fejlesztése, tesztelése, hibakeresése (debug) Xilinx Vivado SDK (Software Development Kit) használatával
8. HW-SW rendszerek együttes tesztelése (Xilinx ChipScope)
9. Egyedi hardver szellemi termékek fejlesztése és tesztelése (ZYBO video/ audio vezérlő)
10. Beágyazott operációs rendszer I.: ARM/MicroBlaze szoft-processzoron Linux rendszer beállítása és indítása
11. Beágyazott operációs rendszer II.: Alkalmazás fejlesztés, tesztelés, meghajtóprogramok, és boot-olás

Ajánlott és felhasznált irodalom

- Fodor Attila, Dr. Vörösházi Zsolt: Beágyazott rendszerek és programozható logikai alkatrészek (TÁMOP 4.1.2) Egyetemi jegyzet (2011)
- 🕮 http://www.tankonyvtar.hu/hu/tartalom/tamop425/0008_fodorvoroshazi/Fodor_Voroshazi_Beagy_0903.pdf (**1. Beágyazott rendszerek**, **2.9 Buszok, beágyazott processzorok fejezetrészek áttekintése ajánlott!!**)

- Xilinx Teaching Materials

  🕮 https://www.xilinx.com/support/university/course-materials.html

- Digilent ZYBO FPGA kártya adatlapok

  – ZYBO kártya:

  🕮 https://store.digilentinc.com/zybo-zynq-7000-arm-fpga-soc-trainer-board/

  – PMOD periféria modulok:

  🕮 https://store.digilentinc.com/pmod-modules-connectors/

További irodalom

- Xilinx hivatalos oldala:
  - 🕮 http://www.xilinx.com
- EE Journal – Electronic Engineering:
  - 🕮 http://www.eejournal.com/design/embedded
- EE Times:
  - 🕮 http://www.eetimes.com/design/embedded

Bevezetés

BEÁGYAZOTT RENDSZEREK

Beágyazott Rendszerek

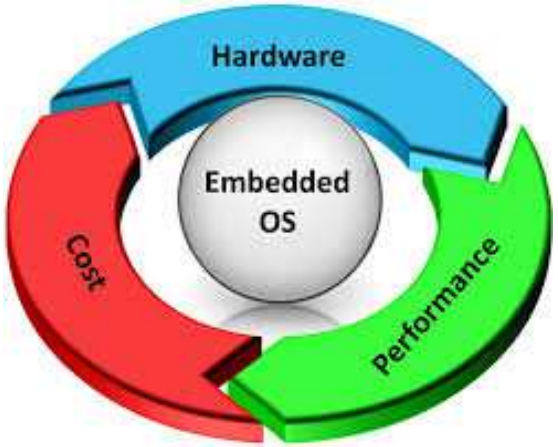
- A beágyazott rendszer (**Embedded System**) a (számítógépes) **hardver-** és **szoftver**elemeknek kombinációja, amely kifejezetten egy adott funkciót, *specifikus* (vezérlési) feladatot képes ellátni, szemben az általános célú számítógép rendszerekkel.
- A beágyazott rendszerek olyan számítógépes eszközöket tartalmazhatnak, amelyek alkalmazás-orientált célberendezésekkel (ASIC, ASSP, **FPGA**, MCU, MPU, DSP, stb.), vagy komplex alkalmazói rendszerekkel (akár OS) szervesen egybeépülve akár azok **autonóm** működését is képesek biztosítani.
- A *programozható* beágyazott rendszerek olyan programozói interfésszel vannak ellátva, amelyek általában sajátos szoftver (firmware) fejlesztési stratégiákat és technikákat követelnek meg.

Néhány fontos alkalmazási terület

- Autóipari alkalmazások: beágyazott elektronikus vezérlők
  - Biztonságkritikus: központi elektronikai vezérlő (ECU), motorvezérlés, fékrásegítő, sebességváltó, blokkolásgátló vezérlés (ABS), kipörgés-gátló (ESP) légzsák
  - Utas központú (komfort) rendszerek: szórakoztatás, ülés/tükör ellenőrzés stb.
- Repülőgép-ipari és védelmi alkalmazások
  - Repülésirányító rendszerek (fedélzeti navigáció, GPS vevő), hajtómű vezérlés, robotpilóta
  - Védelmi rendszerek, radar rendszerek, rádió rendszerek, rakétavezérlő rendszerek
- Gyógyászati berendezések:
  - Orvosi képfeldolgozás
  - Jelmonitorozás (PET, MRI, CT)
- Hálózati/ telekommunikációs rendszerek (modem, router stb.)
- WSN: Vezeték nélküli szenzorhálózatok (motes)
- IoT: Intelligens, vagy smart rendszerek
- Háztartási gépek, ill. fogyasztói elektronika
  - mobiltelefon, PDA, PNA, digitális kamera, nyomtató stb.

Általános követelmények

- Dedikált funkció
  - Jól körülhatárolt (alkalmazás specifikus) funkció(k) támogatása
- Szigorú követelmények
  - Alacsony költség (**C**ost)
  - Gazdaságosság (**E**conomy) - lehetőleg minimális alkatrészből épüljön fel
  - Gyors működés (**S**peed)
  - Alacsony disszipáció (**P**ower)



- Valós idejű (real-time) működés és válasz
  - a környezetet folyamatos monitorozása, és beavatkozás
- Hardver és szoftver részek elkülönült, de együttes tervezése (co-design), tesztelése (co-simulation), ellenőrzése (co-verification)

Alapkövetelmények:

- **Idő**: Egy bekövetkező esemény kezelését a beágyazott rendszer egy meghatározott időn belül kezdje el.
- **Biztonság**: olyan rendszer vezérlése, amely hibás működés esetén egészségkárosodás, és komoly anyagi kár nélkül kezeli a bekövetkező eseményt.

E filozófia mentén a beágyazott rendszerek kettő *alcsoportját* lehet definiálni:

- **Valós idejű rendszer (v. idő kritikus)**: melynél az időkövetelmények betartása a legfontosabb szempont,
- **Biztonságkritikus rendszer**: melynél a biztonsági funkciók sokkal fontosabbak, mint az időkövetelmények betartása.

Megjegyzés: A valóságban nem lehet ilyen könnyedén a beágyazott rendszereket csoportosítani, mert lehetnek olyan valós idejű rendszerek is, melyek rendelkeznek a biztonságkritikus rendszerek bizonyos tulajdonságaival. Szabványok és a törvények szabályozzák azt, hogy milyen alkalmazásoknál kell kötelezően biztonságkritikus rendszert alkalmazni (pl. ADAS ISO 26262).

Valós-idejű rendszerek

A követelmények szigorúsága alapján kétféle valós-idejű (real-time) rendszert különböztethetünk meg:

- **hard real-time rendszer**: *szigorú* követelmények vannak előírva, és a *kritikus* folyamatok meghatározott *időn* belül kell, hogy feldolgozásra kerüljenek,
- **soft real-time rendszer**: a követelmények kevésbé szigorúak, és a *kritikus* folyamatokat a rendszer mindössze nagyobb *prioritással* dolgozza fel.

Ütemezés (scheduling)

A (valós-idejű) operációs rendszerek (**OS**/**RTOS**) számára is kritikus feladat az ütemezés és az *erőforrásokkal való optimális gazdálkodás*. Mivel minden rendszer, valamilyen periféria segítségével kommunikál a környezetével, ezért fontos a perifériák valós idejű rendszer követelményeinek megfelelő módon történő kezelése: a válaszidő betartásához az eseményt lekezelő *utasítás sorozatot* végre kell hajtani. Az utasítássorozat lefutása *erőforrásokat* igényel, melyeket az operációs rendszernek kell biztosítani, hogy hozzá tudja rendelni az időkritikus *folyamatokhoz*.

A processzorok **ütemezésének** következő szintjeit lehet megkülönböztetni:

- Hosszú-távú (long term) ütemezés vagy munka ütemezés,
- Közép-távú (medium term) ütemezés,
- Rövid-távú (short term) ütemezés.

Ütemezés szintjei

Az operációs rendszerek magja (kernel) tartalmazza az ütemezőt.

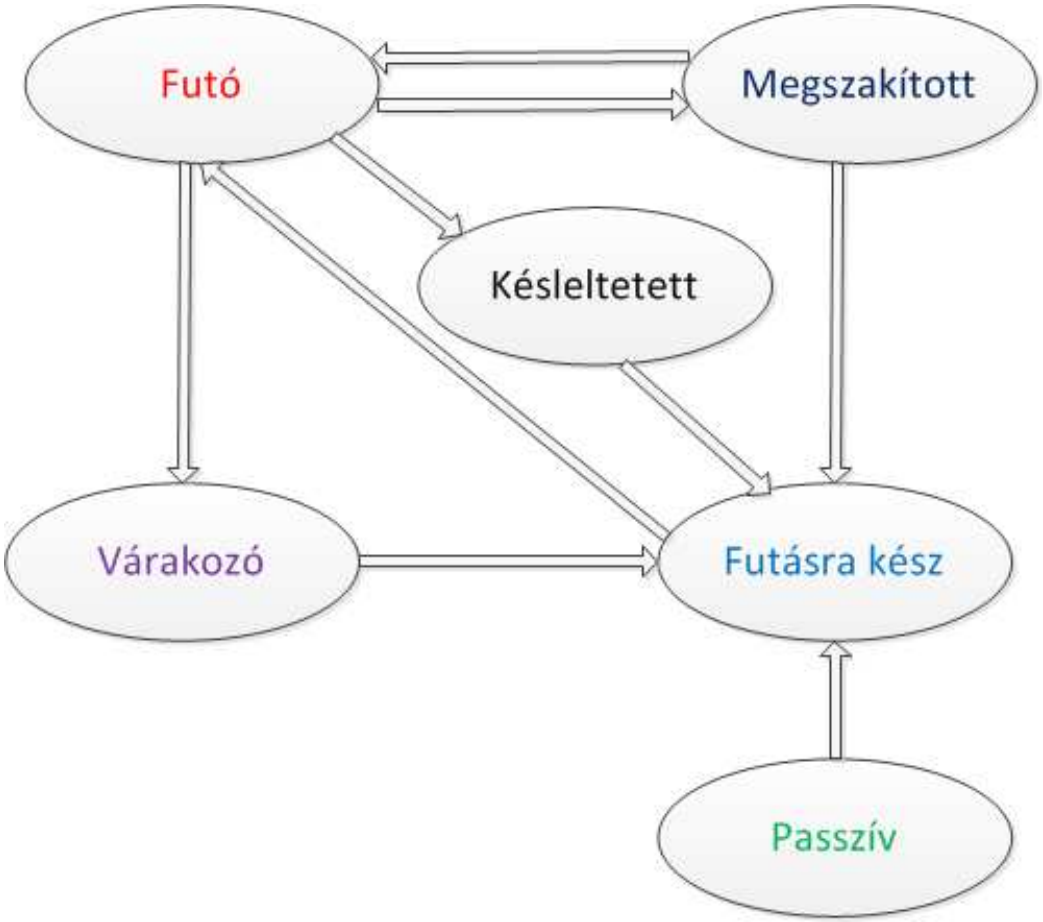
- **A hosszú-távú ütemezés** feladata, hogy a *háttértáron* várakozó, még el nem kezdett munkák közül meghatározza, melyek kezdjenek el futni, a munka befejeződésekor ki kell választania egy új elindítandó munkát. A hosszú-távú ütemezést végző algoritmusnak ezért *ritkán* kell futnia.
- **A közép-távú ütemezés** az időszakos *terhelésingadozásokat* hívatott megszüntetni, hogy a nagyobb terhelések esetében ne legyenek időtúllépések. A középtávú ütemező algoritmus ezt úgy oldja meg, hogy bizonyos (nem időkritikus) folyamatokat *felfüggeszt*, majd *újraaktivál* a rendszer terhelésének a függvényében. Folyamat felfüggesztése esetén a folyamat a *háttértáron* tárolódik, az operációs rendszer elveszi a folyamattól az erőforrásokat, melyeket csak a folyamat újraaktiválásakor ad vissza a felfüggesztett folyamatnak.
- **A rövid-távú ütemezés** feladata, hogy kiválassza, hogy melyik futásra kész folyamat *kapja meg a processzort*. A rövidtávú ütemezést végző algoritmus *gyakran* és *gyorsan* fut le, ezért az operációs rendszer mindig a *memóriában* tartja az ütemező kódját.

Ütemezés – további fogalmak

Az ütemezéssel és a programokkal kapcsolatban a következő alapfogalmak értelmezhetők:

- **Task**: Önálló részfeladat.
- **Job**: A task-ok kisebb, rendszeresen végzett részfeladatai.
- **Process**: A legkisebb futtatható programegység, egy önálló ütemezési entitás, amelyet az OS önálló programként kezel. Van saját (védett) memória területe, mely más folyamatok számára elérhetetlen. A task-okat folyamatokkal implementálhatjuk.
- **Thread**: Saját memóriaterület nélküli ütemezési entitás, az azonos szülőfolyamathoz tartozó *szálak* azonos memóriaterületen dolgoznak.
- **Kernel**: Az operációs rendszer alapvető eleme, amely a task-ok kezelését, ütemezést és a task-ok közti kommunikációt biztosítja. A kernel kódja hardver-függő (device driver), valamint hardverfüggetlen rétegekből együttesen épül fel.

Task állapotok változása



- **Passzív (Dormant)**: Passzív (nyugvó) állapot, amely jelentheti az inicializálás előtti vagy felfüggesztett állapotot.
- **Futásra kész (Ready)**: A futásra kész állapotot jelöli. Fontos a task prioritási szintje és az is, hogy az éppen aktuálisan futó task milyen prioritási szinttel rendelkezik, ezek alapján dönti el az ütemező, hogy elindítja e a taskot.
- **Futó (Running)**: A task éppen tevékenyen fut.
- **Késleltetett (Delayed)**: Ez az állapot akkor lép fel, mikor a task valamilyen *időintervallumig* várakozni kényszerül. Rendszerint szinkron időzítő (*timer*) szolgáltatás hívása után következik be.
- **Várakozó (Waiting)**: A task egy meghatározott eseményre várakozik. (Ez rendszerint valamilyen I/O periféria művelet szokott lenni.)
- **Megszakított (Interrupted)**: A task-ot megszakították, vagy a megszakítás kezelő rutin éppen megszakítja a folyamatot (IRQ, INT).

Ütemezési algoritmusok

Az ütemezési algoritmusoknak két fő típusa van:

- **Kooperatív** (=nem preemptív): A működési elve és alapötlete, hogy egy adott program vagy folyamat *lemond* a processzorról, ha már befejezte a futását vagy valamilyen I/O műveletre vár. Ez az algoritmus addig működik jól és hatékonyan, amíg a szoftverek megfelelően működnek (nem kerülnek végtelen ciklusba) és *lemondanak* a processzorról. Ha viszont valamelyik a program/folyamat nem mond le a processzorról vagy kifagy, akkor az egész rendszer stabilitását képes lecsökkenteni. A kooperatív algoritmus ezért soha nem fordulhat elő valós-idejű beágyazott operációs rendszerek esetében.
- **Preemptív**: az operációs rendszer részét képező *ütemező algoritmus vezérli* a programok/folyamatok futását. A preemptív multitask esetén az operációs rendszer *elveheti* a folyamatoktól a *futás jogát* és átadhatja más folyamatoknak. A valós idejű operációs rendszerek ütemezői minden esetben preemptív algoritmusok, így bármely program vagy folyamat leállása nem befolyásolja számottevően a rendszer stabilitását.

Task-ok közötti kommunikáció

Mivel a rendszer működése közben a task-ok egymással párhuzamosan futnak ezért gondoskodni kell arról, hogy egyazon I/O perifériát, erőforrást vagy memória területet két vagy több task ne használjon egyszerre, mert abból hibás rendszerműködés alakulna ki.

A következő ismert módszerek állnak rendelkezésre:

- **Mutex (kölcsönös kizárás):** ún. „locking” mechanizmus (csak a task amelyik zárolta, oldhatja fel)
- **Szemafor (semaphore)**: „signaling” mechanizmus (egyik task jelez a másiknak, hogy végzett, és átveheti az erőforrást) ~ 1 bit információ
- **Események (event flags)**: melyek több bit információ kicserélésére is alkalmasak.
- **Postaláda (mailbox):** amely akár komplexebb adatstruktúra átadására is szolgálhat.
- **Sor (queue):** amely több mailbox tömbjében lévő tartalom átadására szolgál.
- **Cső (pipe vagy FIFO):** amely direkt, folyamatos (akár streaming) kommunikációt tesz lehetővé két task között.

(Beágyazott) Operációs rendszerek

Többféle csoportosítás lehetséges:

- Általános célú, vagy **beágyazott OS**
- Valós-idejű (idő-kritikus), vagy nem-időkritikus
- Nyílt forráskódú, vagy licenszelhető, stb.

Általános célú processzorok operációs rendszerei (OS):

- MS-DOS, Linux, Windows, stb.

Beágyazott processzorok *valós-idejű* operációs rendszerei (RTOS):

- Linux
- Android
- Micrium uC/OS
- QNX
- RTLinux
- **Windriver VxWorks (RT)**
- Windows Embedded, IoT, stb...

Processzorok osztályozása

- Integráltság szerint:
  - uP/CPU: hagyományos mikroprocesszorok + fizikailag különálló memória + külső I/O periféria chipek (chipset)
  - uC/MCU: mikrokontrollerek: egyetlen chipen integrálva a processzor, a memória (ált. flash), és néhány I/O periféria
    - System-on-a-Chip (SoC) : egychipes rendszer
    - Kis méret és költség, alacsony disszipált teljesítmény
- Utasítás készlet szerint:
  - RISC vs. nem RISC (CISC) ISA – utasításkészletű architektúrák
- Utasítás / Adat memória hozzáférés szerint:
  - Von Neumann (közös) vs. Harvard architektúrák (elkülönült)

Néhány architektúra típus: Intel 8051, ARM, AVR, MicroChip(PIC), MIPS, IBM PowerPC, x86 (32/64), Sun SPARC, stb.

Fogalmak tisztázása

- **FPGA**: általában logikai és dedikált erőforrásokból álló programozható áramkör
  - Pl. Xilinx Artix-7 sorozata
- **SoC**: **S**ystem-**o**n-a-**C**hip = egychipes rendszer/számítógép
  - Minden funkciót (analóg, digitális, vagy RF) akár egyetlen chipre integrálnak, ahelyett, hogy sok különböző eszközt használnának. Igaz ez egy mai MCU, DSP, ASIC, vagy FPGA esetében is.

- **Zynq** = *„Zync" – cink*, mint ötvöző elem. Szorosan integrálja a hagyományos FPGA logikát (PL) a processzor rendszerrel (PS) => PL+PS integráció
- **APSoC** (lényegében a Xilinx Zynq ilyen): **A**ll **P**rogrammable **SoC,** azaz minden komponensében programozható SoC chipet jelent.
- Tehát a Zynq APSoC egy olyan chip, amelyben két rész van integrálva
  - Hagyományos FPGA logika (PL) = Artix-7 FPGA logika, illetve
  - Processzor rendszer (PS): ARM-Cortex-A9 magok

Technológiák és stratégiák

Élenjáró *technológiák* a beágyazott rendszerek tervezéséhez és megvalósításához – processzáló egységek csoportosítása:

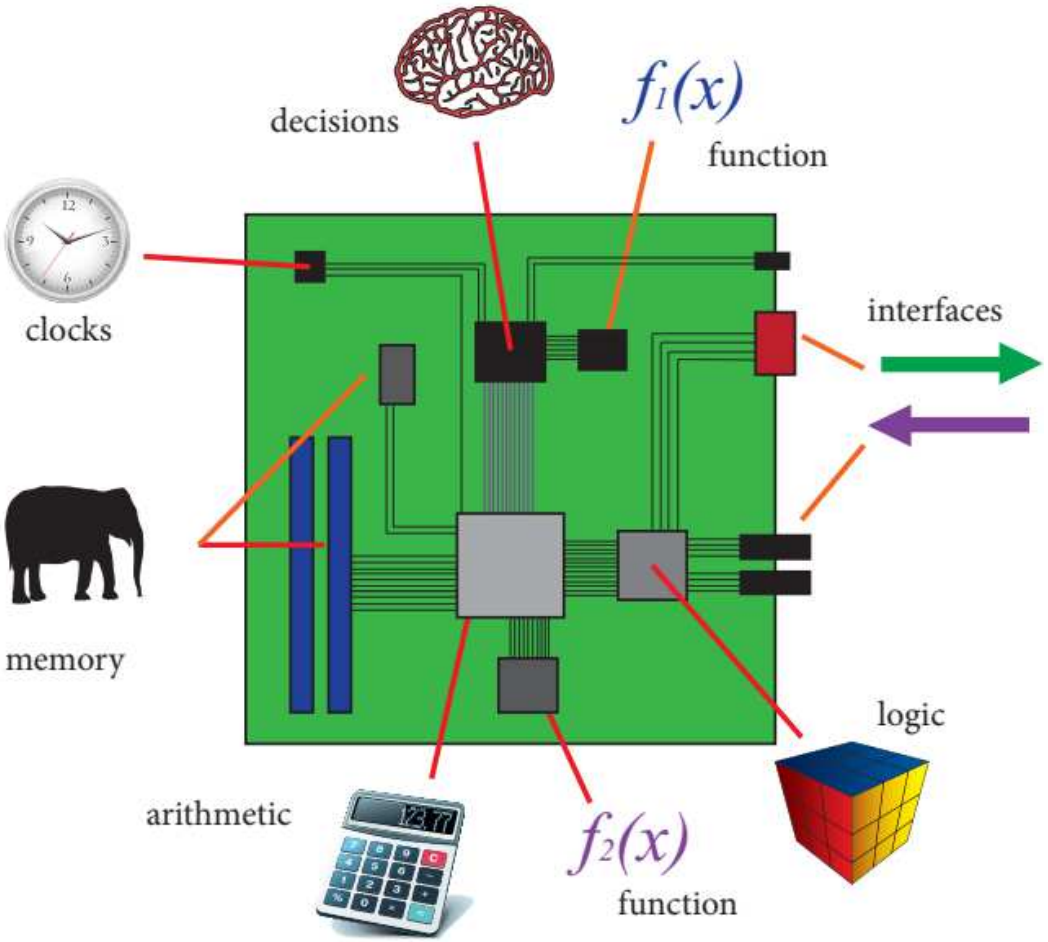
- ***(DSP):*** Digitális jelfeldolgozó processzor alapú rendszerek
- ***(MCU):*** Mikrovezérlő-alapú rendszerek
- ***(ASIC/ASSP):*** Alkalmazás specifikus (berendezés orientált) integrált áramköri technológián alapuló rendszerek
- ***(FPGA):*** Programozható logikai kapuáramkörök technológián alapuló rendszerek
- **(MPU/GPU):** Mikroprocesszor, vagy grafikus processzor

- **SoC**: **System-on-a-Chip**: olyan egychipes rendszer, amely a fenti technológiákat akár együtt integrálva is tartalmazhatja!

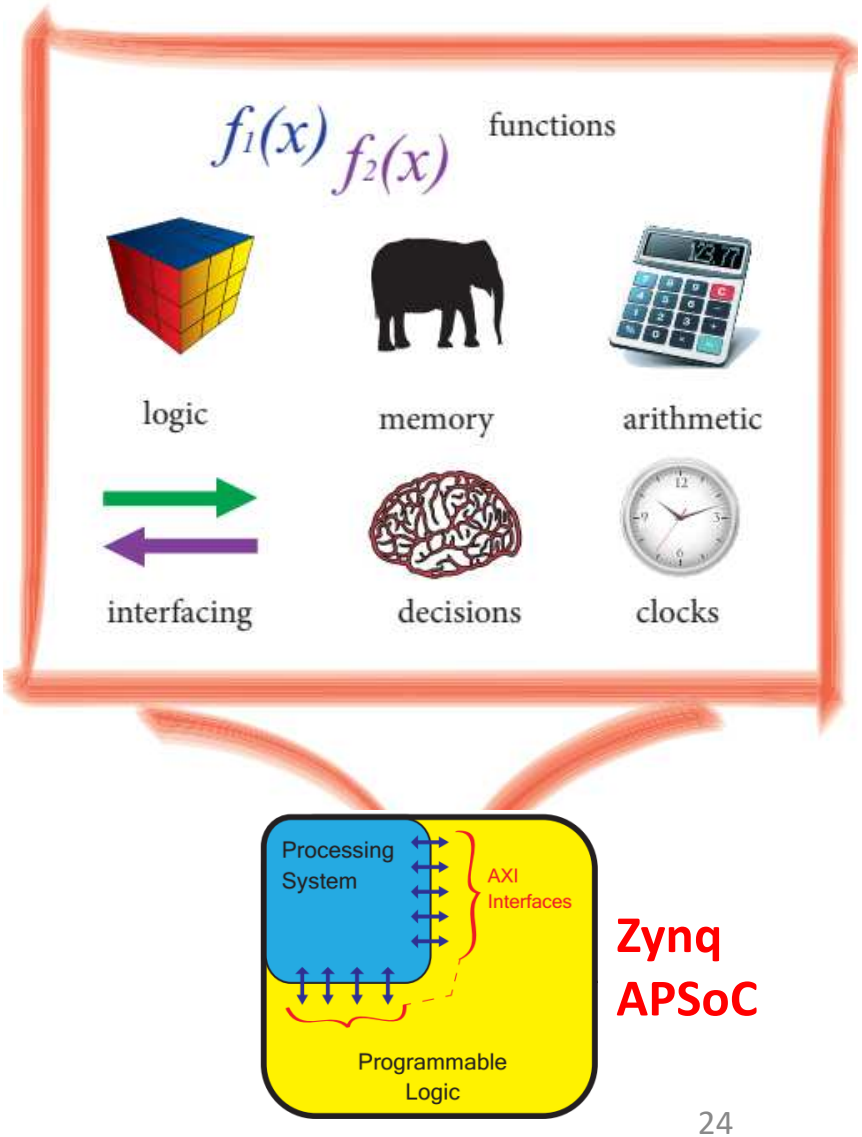
Fejlesztési *stratégiák*:

- HW/SW co-design: HW/SW részek együttes tervezése
- HW/SW co-verification: HW/SW részek együttes ellenőrzése és tesztelése

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



vs.



Zynq APSoC

I/O Perifériák

- Aszinkron soros kommunikációs interfészek: **RS-232**, RS-422, RS-485, stb.
- Szinkron Soros kommunikációs interfészek: **I²C, SPI** stb.
- Univerzális soros busz: **USB**
- Multimédia kártyák: (SD) Smart Cards, (CF) Compact Flash stb.
- Hálózat: Ethernet (1GbE / 10 GbE / 100 GbE)
- Ipari hálózati un. „Field-bus” protokollok: CAN, LIN, PROFIBUS, stb.
- Időzítő-ütemezők: PLL(s), Timers, Counters, Watchdog timers (WDT)
- Általános célú I/O-k (General Purpose I/O - **GPIO**): LED-ek, nyomógombok, kapcsolók, LCD kijelzők, stb.
- Analóg-Digitális/Digitális-Analóg (ADC/DAC) konverterek
- Debug portok: **JTAG**, ISP, ICSP, BDM, DP9, stb.

FPGA alapú beágyazott rdsz.

FPGA-alapú beágyazott rendszerek főbb tervezési lépései:

- FPGA hardver (firmware) tervezés,
  - Beágyaz**ható**/beágyaz**ott** processzor kiválasztása:
    - Licenszelhető Soft-core: PicoBlaze / MicroBlaze™ (Xilinx); Nios II™ (Altera), …
    - Licenszelhető Hard-core: IBM PowerPC® (Xilinx), ARM® (Xilinx / Altera), …
    - Nyílt forráskódú processzor magok: pl. www.opencores.org
- Programozható Perifériák kiválasztása (lásd. a jegyzet *Fejlesztő kártyák*, ill. *Beágyazott perifériák* részei),
- Eszközmeghajtók (driver) és szoftver könyvtárak (lib) generálása
  - **BSP**: **B**oard **S**upport **P**ackage,
- Alkalmazás fejlesztés:
  - Szoftver rutinok (API),
  - Megszakítás-kezelő rutinok,
  - Operációs rendszer, valós-idejű operációs rendszer.

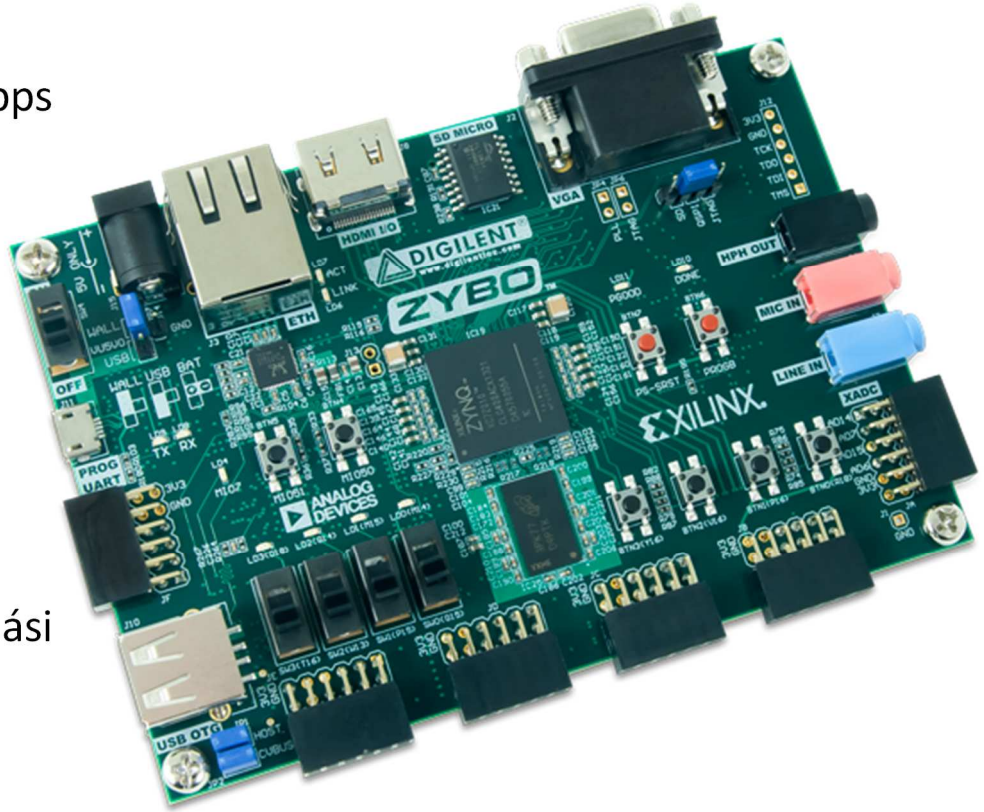
Használt fejlesztő Hardverek rövid bemutatása

DIGILENT ZYBO + PMOD

Digilent ZYBO fejlesztő kártya

ZYBO™ Zynq FPGA/APSoC fejlesztő kártya

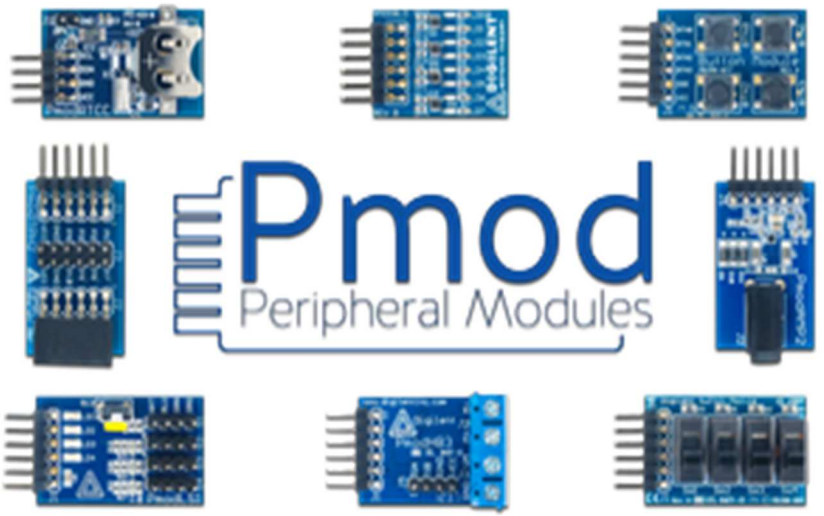
- *Xilinx Zynq-7000 (Z-7010)*
  - 650 MHz dual ARM Cortex-A9 magok (PS)
  - 8-csatornás DMA vezérlő (PS)
  - 1G ethernet, I2C, SPI, USB-OTG vezérlő (PS)
  - Artix-7 FPGA logika (PL)
  - 28K logikai cella, 240 Kbyte BRAM, 80 DSP szorzó(PL)
  - 12-bites, 1MSPS XADC (PL)
- 512 Mbyte DDR3 x32-bit (adatbusz), 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)



📖 Weblap: https://store.digilentinc.com/zybo-zynq-7000-arm-fpga-soc-trainer-board/

Bővítési lehetőség - PMOD

- Periféria modulok (PMOD), további bővítési lehetőségek
  - Karakteres LCD, OLED, 7segLEG
  - GPS vevő, WiFi, Bluetooth,
  - Ethernet IF, USB-UART, RS232
  - Joystick, Rotary Enc., Switches,
  - SD Card, Serial Flash,
  - A/D, D/A konverterek, H-hidak
  - Gyorsulás-mérő, Perdület-mérő (Gyroscope),
  - Hőmérséklet-mérő, ...stb.



vagy

  - „3rd party” gyártók, esetleg egyedi tervezésű bővítőkártyák fejlesztése, és illesztése.

📖 Weblap: https://store.digilentinc.com/pmod-modules-connectors/

Xilinx Vivado 2018.3

HASZNÁLT FEJLESZTŐ SZOFTVEREK

Fejlesztő szoftverek

Milyen fejlesztő szoftver eszközökkel ismerkedünk meg a félév során:

- *Xilinx Embedded System Design Flow Workshop / Teaching Materials* segédanyagai alapján
  - (eléréséhez regisztráció szükséges - ingyenes)
- **Xilinx Vivado Design Suite (System Edition Full) 2018.3**
  - **Vivado Embedded Development Kit (~EDK)**
  - **Vivado Software Developer Kit (SDK)**
- **Digilent Adept Suite:**
  - konfigurációs bitfájl letöltéséhez (csak a fejlesztő kártya teszteléséhez)
- **Xilinx ISE Development Suite (ISE)**
  - komponensek fejlesztése magas-szintű hardver leíró nyelven (HDL)
- **Xilinx ISim (Simulation)**
  - komponensek szimulációs tesztje
- **Xilinx Embedded Development Kit (EDK)**
  - **Xilinx Platform Studio (XPS)** v.14.7 használata
    - Xilinx FPGA alapú beágyazott rendszer (firmware) fejlesztő környezete
  - **Xilinx Software Developer Kit (SDK)** v.14.7 használata
    - Xilinx FPGA beágyazott szoftver-rendszer fejlesztő környezete
- Xilinx ChipScope v.14.7 használata:
  - digitális logikai analizátor (Xilinx JTAG-USB)