BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR
MÉRÉSTECHNIKA ÉS INFORMÁCIÓS RENDSZEREK TANSZÉK

Mikrorendszerek tervezése

Vivado fejlesztői környezet

Fehér Béla
Raikovich Tamás

Vivado fejlesztői környezet

Integrált fejlesztői környezet a Xilinx 7-es sorozatú vagy újabb eszközökhöz

ISE Design Suite

- **ISE: (elsősorban) HDL alapú fejlesztés**
- **EDK: IP alapú fejlesztés (processzoros rendszerek)**
- **ChipScope: belső logikai analizátor**
- **SDK: szoftverfejlesztés**

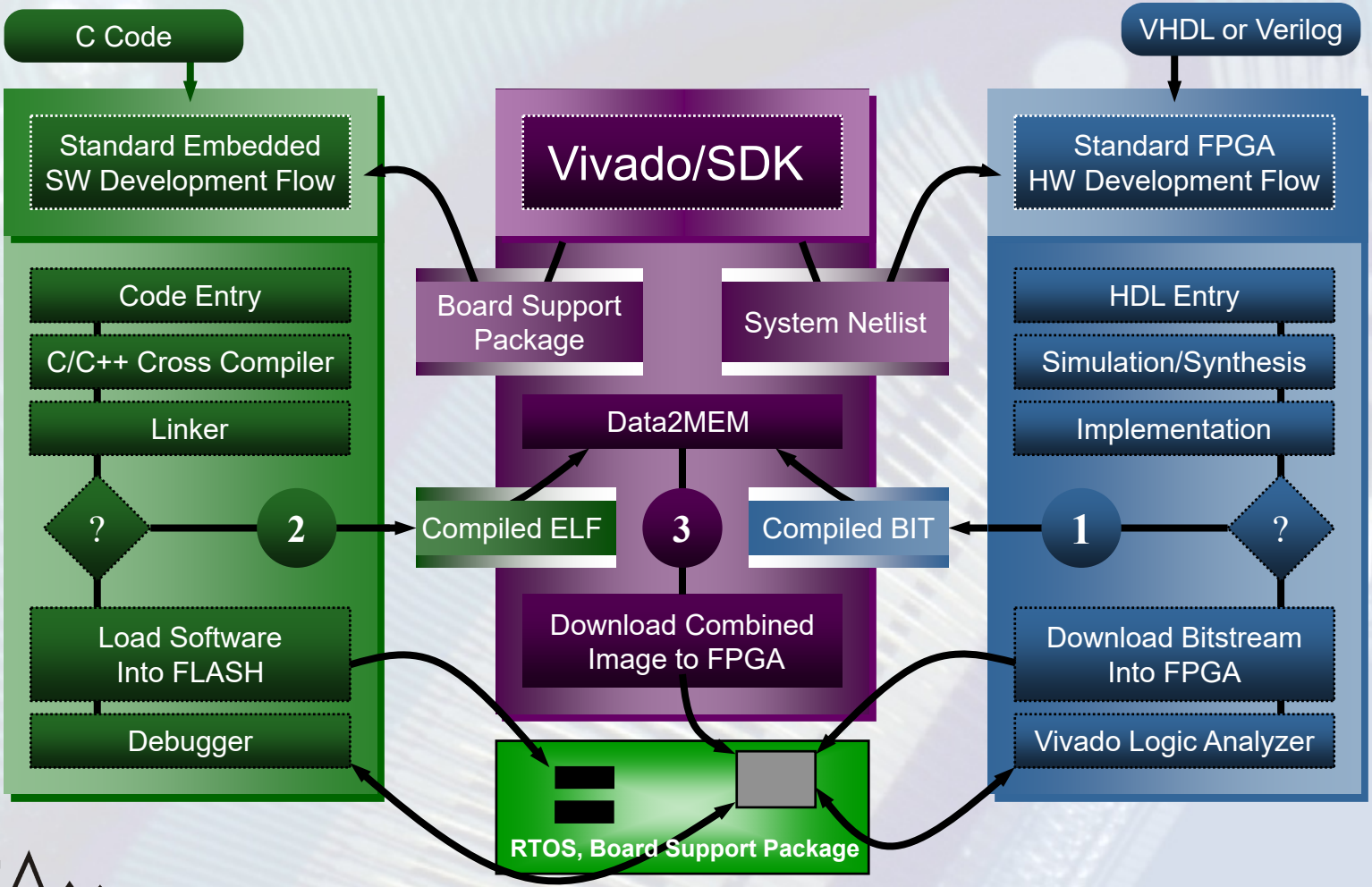
Vivado Design Suite

- **Vivado: HW fejlesztés**
  - HDL alapú fejlesztés
  - IP Integrator
  - Vivado Logic Analyzer
- **SDK[1], Vitis[2]: szoftverfejlesztés**

1: Vivado 2019.1 verzióig
2: Vivado 2019.2 verziótól

Vivado fejlesztői környezet

A fejlesztési folyamat áttekintése:



Vivado fejlesztői környezet

Beágyazott hardver fejlesztés:

- **Projekt létrehozása**
- **Block Design (és HDL forrásfájlok) létrehozása**
  - Gyári perifériák hozzáadása
  - Saját periféria létrehozása, hozzáadása
  - Vivado logikai analizátor beillesztése
- **Szimuláció**
- **Szintézis, felhasználói megkötések megadása (XDC)**
- **Implementáció, FPGA konfigurációs fájl generálás**

Vivado fejlesztői környezet

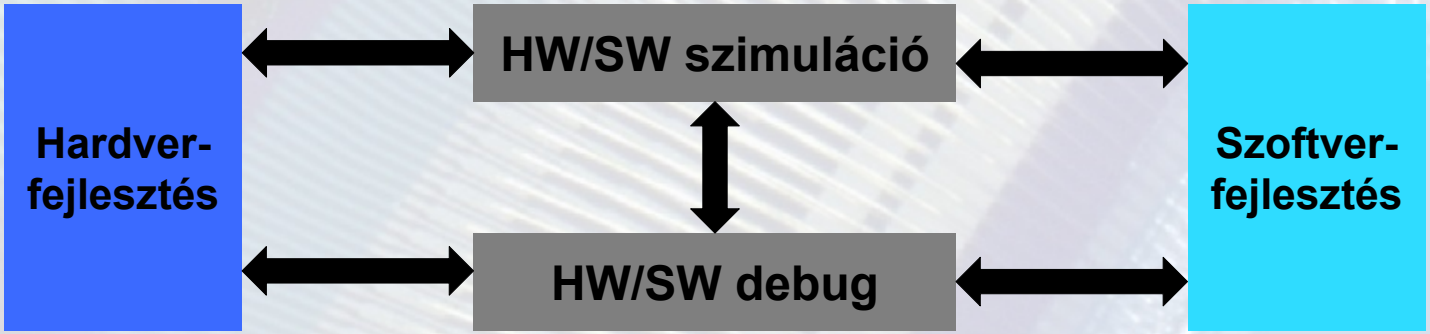
Beágyazott szoftver fejlesztés:

- **A hardver információk exportálása**
- ***Software Development Kit (SDK), Vitis***
  - Hardver platform projekt
  - Board support package (BSP)
    - Eszközmeghajtók a perifériákhoz, szoftverkönyvtárak
    - Operációs rendszer: standalone (nincs OS), FreeRTOS
  - Szoftver projekt
  - Kapcsolódás a célrendszerhez
    - XSCT (Xilinx Software Command-line Tool) server, konzol
  - Debug

Vivado fejlesztői környezet

Hardver és szoftver integrálás:

- **A bitfolyam létrehozása és az FPGA konfigurálása**
  - A belső Blokk-RAM-ok tartalmának frissítése a futtatható kóddal
- **Külső flash memória konfigurálása**
  - Vivado Hardware Manager
  - SDK, Vitis: Create Boot Image / Program Flash



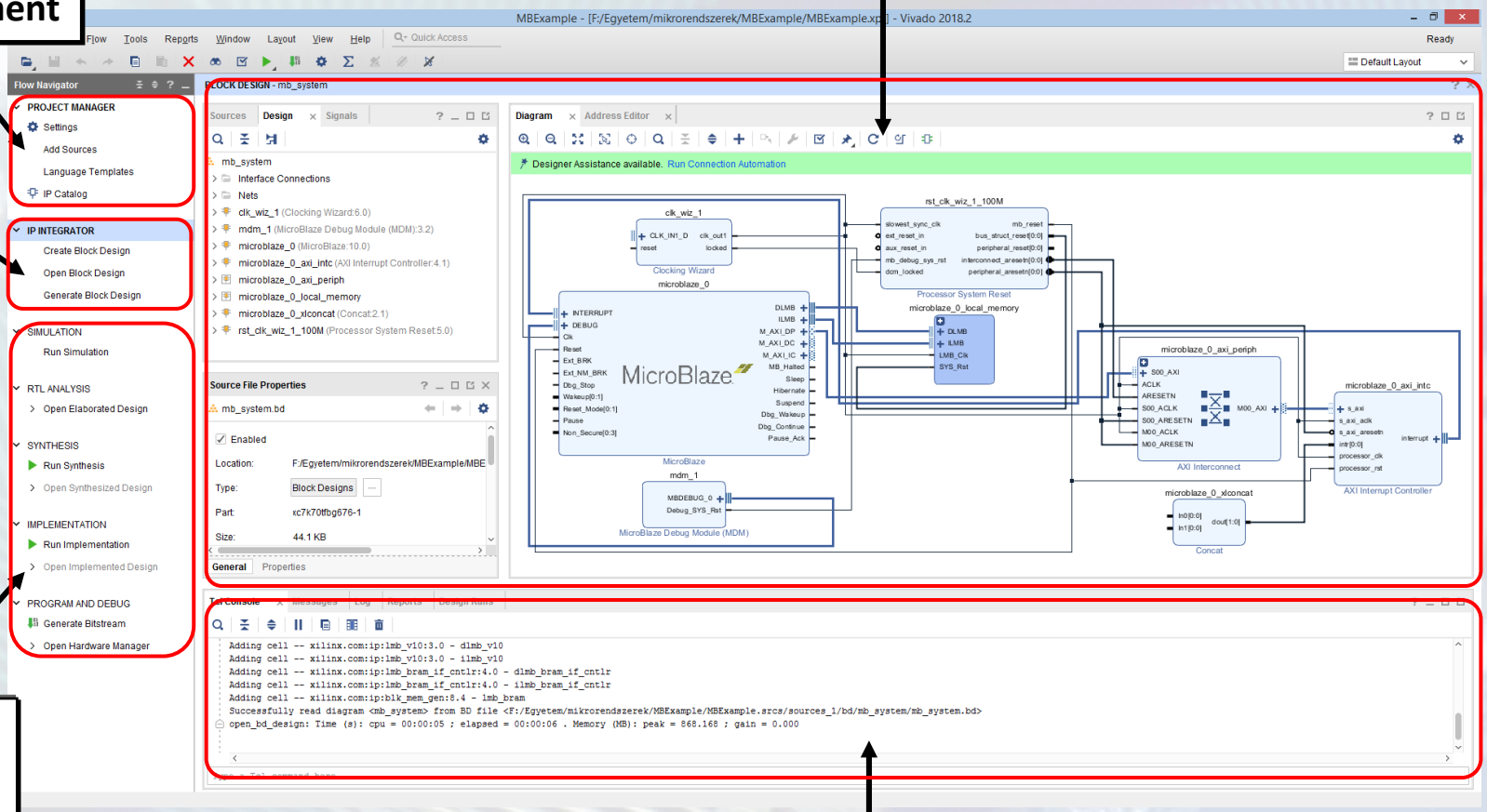
Vivado fejlesztői környezet

Projekt menedzsment

Projekt nézet panel

IP Integrator

FPGA fejlesztési folyamat



Konzol, üzenetek, log-ok

Projekt létrehozása

1. **A projekt nevének és helyének megadása**
2. **A projekt típusának megadása**
   - ***RTL Project:*** HDL és Block Design források használata, szintézis és implementációs lépések hajthatóak végre
   - ***Post-synthesis project:*** huzalozási listák használata, az implementációs lépések hajthatóak végre
   - ***I/O Planning project:*** a kiválasztott eszköz erőforrásai tekinthetőek meg, nincsenek forrásfájlok
   - ***Imported project:*** korábbi ISE, XST, Synplify projekt importálása
   - ***Example project:*** új projekt létrehozása előre definiált sablon alapján

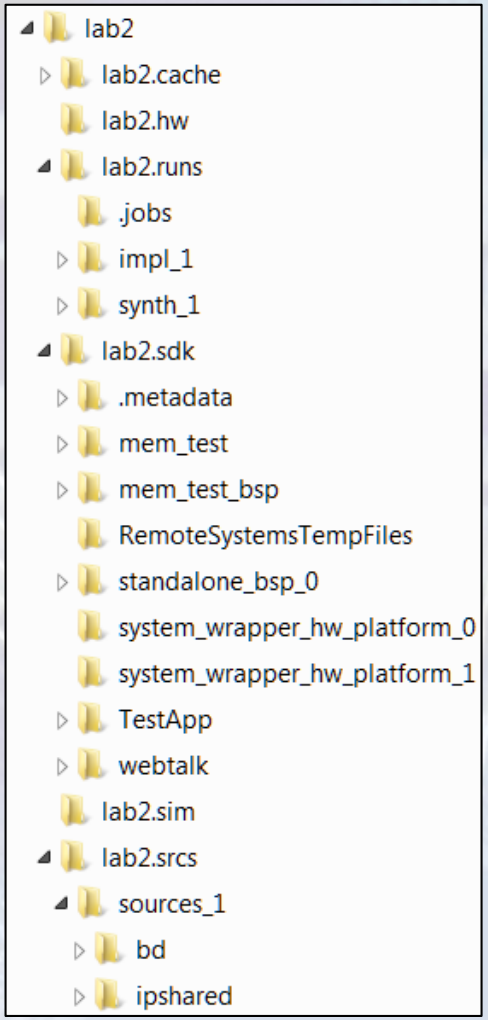
Projekt létrehozása

3. **Opcionálisan már meglévő forrásfájlok hozzáadása vagy új forrásfájlok létrehozása**
4. **A felhasznált eszköz vagy fejlesztői kártya kiválasztása**
   - Utóbbi esetén szükséges a kártyához tartozó leíró fájl
   - Bővített lehetőségek állnak rendelkezésre a kártyán és a processzoros rendszerben lévő perifériák összekapcsolásánál (pl. GPIO port ↔ LED-ek, kapcsolók)
5. **Összegzés az új projektről**
6. **Létrejön a projekt**

Projekt létrehozása

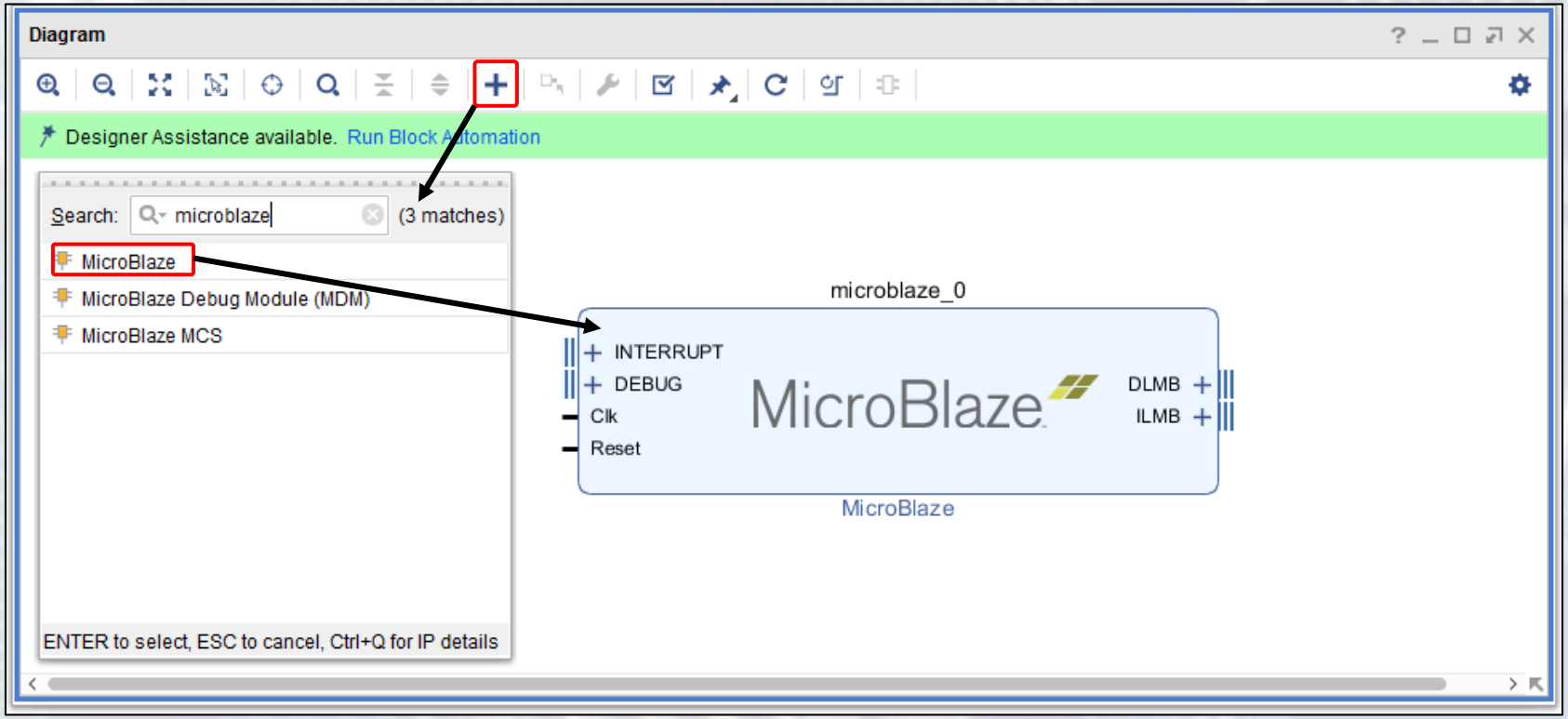
Vivado projekt struktúra:

- .xpr Vivado projekt fájl
- *.srcs könyvtár*: projekt forrásfájlok, IP Integrator fájlok helye
- *.sim könyvtár*: szimulációval kapcsolatos fájlokat tartalmazza
- *.runs könyvtár*: a szintézis és az implementáció során létrejött fájlok
- *.sdk könyvtár*: HW platform és SDK szoftver projekteket tartalmazza
- *.cache* könyvtár: ideiglenes fájlok



IP Integrator – Block Design

- IP Integrator → Create Block Design
- Az IP katalógusból húzhatók be az új elemek



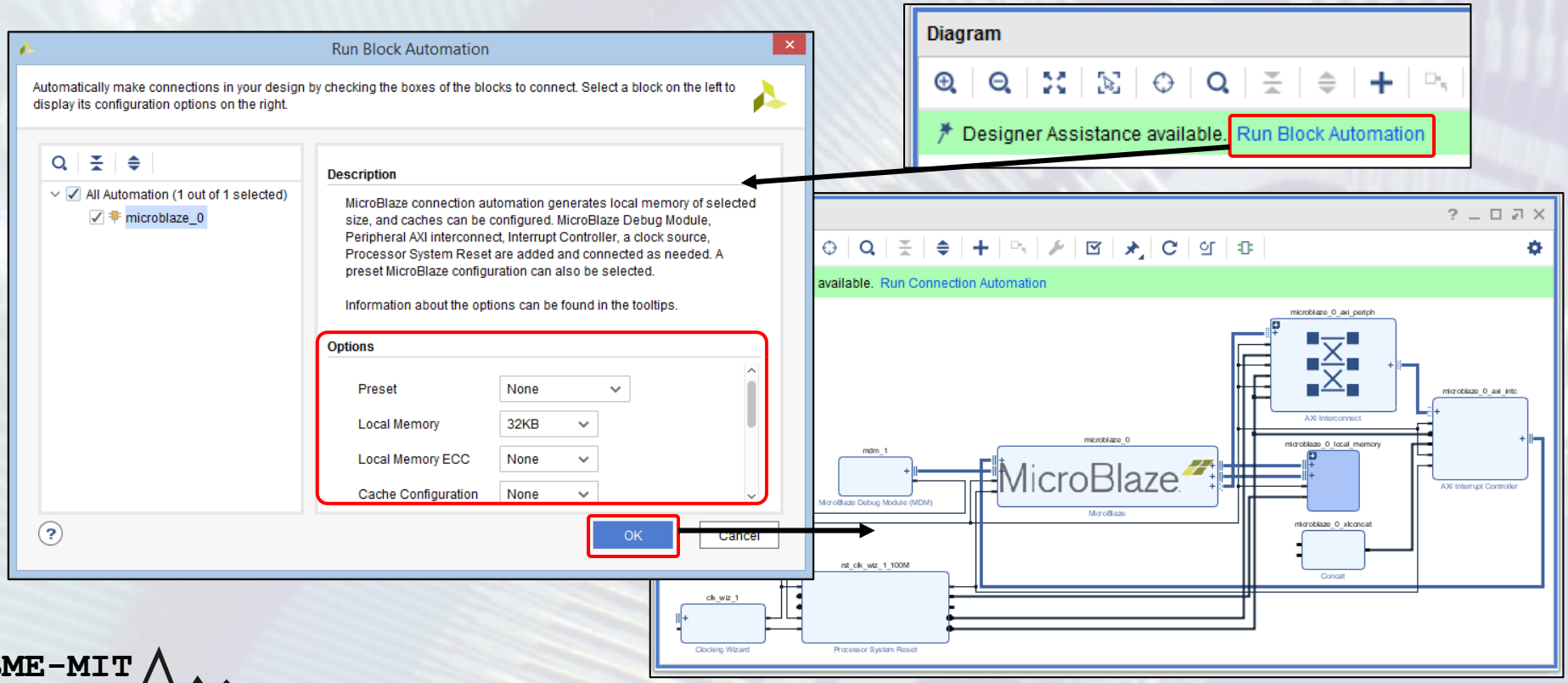
IP Integrator – Block Design

- **Intelligens tervezési környezet**
  - Block Automation
  - Connection Automation
  - Az érvényes csatlakozási lehetőségek kiemelése
  - IP-k csoportosítása, hierarchikus tervek kialakítása
- **IP Packager: egyedi IP-k létrehozása, importálása**
  - IP létrehozása Block Design-ból
  - AXI buszra illeszkedő IP létrehozása
  - A részletekről külön előadás lesz

IP Integrator – Block Design

Tervezési segítség: Block Automation

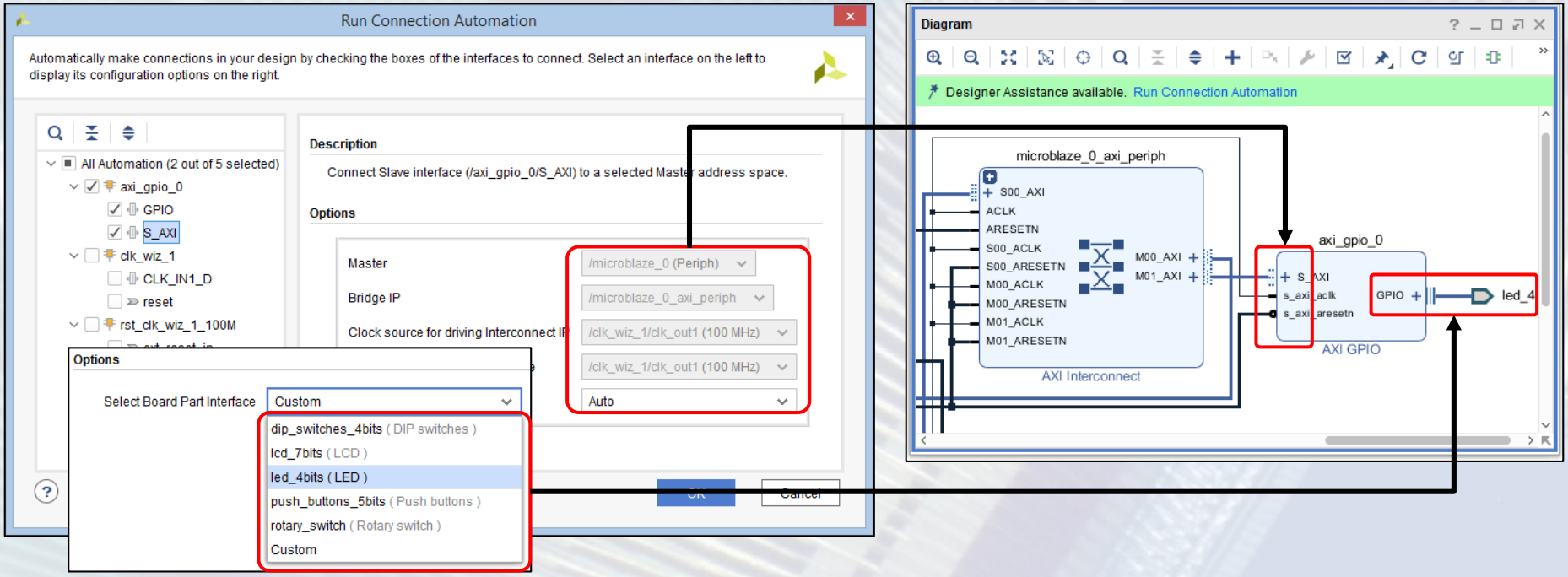
- Előre meghatározott beállítások kiválasztása
- A működéshez szükséges IP-k hozzáadása



IP Integrator – Block Design

Tervezési segítség: Connection Automation

- **Összeköttetések létrehozása a megadott opciók alapján**
- **Portok külső perifériákhoz rendelése (pl. GPIO → LED)**
  - Ha a projekt létrehozásánál fejlesztői kártyát adtunk meg

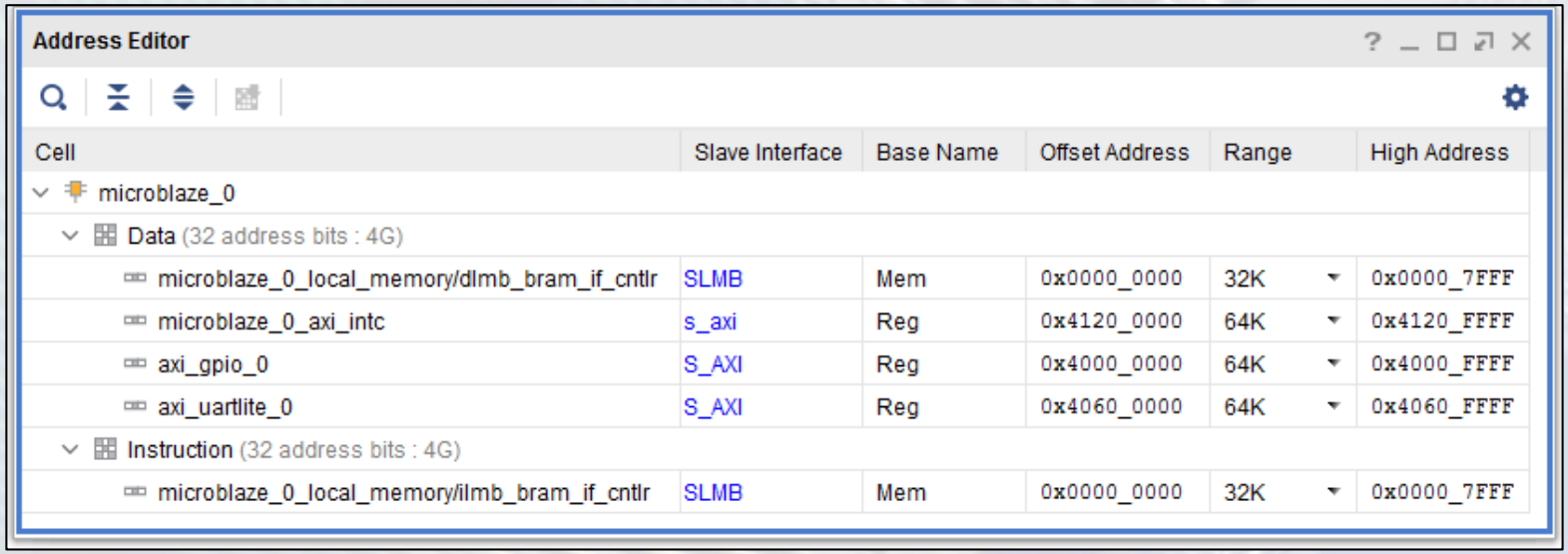


IP Integrator – Block Design

- **IP portok összekötése, bekötése**
  - Az összeköttetés behúzása az egérkurzorral
  - Zöld pipa jelzi a helyes bekötési lehetőségeket
- **Külső port hozzáadása**
  - Jobb klikk → Create Port
- **IP port kivezetése külső portra**
  - Jobb klikk → Make External
- **Properties panel: a kiválasztott elem tulajdonságai**
  - Név megváltoztatása
  - Órajel portnál az órajel frekvencia megadása
  - Reset portnál a polaritás megadása

IP Integrator – Block Design

- **IP beállítások módosítása**
  - Dupla kattintás az adott blokkon
- **Blokk diagram újragenerálása (áttekinthetőség javítása)**
  - Toolbar → Regenerate Layout
- **Periféria címek beállítása: Address Editor**



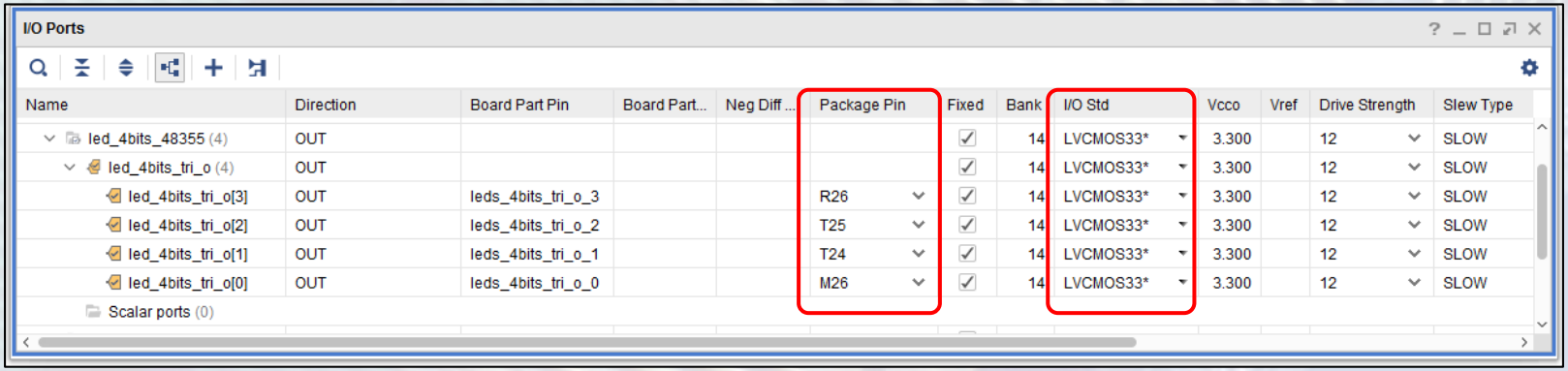
Cell	Slave Interface	Base Name	Offset Address	Range	High Address
microblaze_0					
Data (32 address bits : 4G)					
microblaze_0_local_memory/dlmb_bram_if_cntlr	SLMB	Mem	0x0000_0000	32K	0x0000_7FFF
microblaze_0_axi_intc	s_axi	Reg	0x4120_0000	64K	0x4120_FFFF
axi_gpio_0	S_AXI	Reg	0x4000_0000	64K	0x4000_FFFF
axi_uartlite_0	S_AXI	Reg	0x4060_0000	64K	0x4060_FFFF
Instruction (32 address bits : 4G)					
microblaze_0_local_memory/ilmb_bram_if_cntlr	SLMB	Mem	0x0000_0000	32K	0x0000_7FFF

IP Integrator – Block Design

- **Meglévő Verilog modul hozzáadása a Block Design-hoz**
  - Jobb klikk → Add Module…
- **Block Design ellenőrzése**
  - Toolbar → Validate Design
  - A hibákat és a kritikus figyelmeztetéseket javítsuk ki
  - Bizonyos módosítások csak az ellenőrzés után jelennek meg a diagramon
- **HDL Wrapper hozzáadása**
  - Sources panel → jobb klikk a Block Design-on → Create HDL Wrapper…
  - A kétirányú portok és a háromállapotú kimenetek esetén itt kerülnek példányosításra az I/O bufferek

Megkötések megadása

- **A szintézis futtatása és a szintetizált rendszer betöltése után megadhatóak a megkötések**
  - FPGA lábak hozzárendelése a top-level portokhoz
  - Időzítési paraméterek, pl. órajel frekvencia
- **Window menü → I/O ports**
  - ***FPGA láb***, ***I/O szabvány***, meghajtás erősség, stb.



Name	Direction	Board Part Pin	Board Part...	Neg Diff ...	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type
led_4bits_48355 (4)	OUT					✓	14	LVCMOS33*	3.300		12	SLOW
led_4bits_tri_o (4)	OUT					✓	14	LVCMOS33*	3.300		12	SLOW
led_4bits_tri_o[3]	OUT	leds_4bits_tri_o_3			R26	✓	14	LVCMOS33*	3.300		12	SLOW
led_4bits_tri_o[2]	OUT	leds_4bits_tri_o_2			T25	✓	14	LVCMOS33*	3.300		12	SLOW
led_4bits_tri_o[1]	OUT	leds_4bits_tri_o_1			T24	✓	14	LVCMOS33*	3.300		12	SLOW
led_4bits_tri_o[0]	OUT	leds_4bits_tri_o_0			M26	✓	14	LVCMOS33*	3.300		12	SLOW
Scalar ports (0)												

Megkötések megadása

- **Tools menü → Timing → Constraints Wizard…**
  - Időzítési megkötések generálása, megadása
- **A megkötéseket az XDC fájlok tárolják**
  - Részben szabványos (SDC, Synopsys Design Constraints)
  - TCL parancsokat tartalmaznak (lásd a példát)
  - Részletek: Vivado User Guide – Using Constraints (UG903)
- **Megkötések automatikus importja a kártya leíró fájlból**

```
set_property IOSTANDARD LVCMOS33 [get_ports {btn_tri_i[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn_tri_i[0]}]
set_property PACKAGE_PIN A17 [get_ports {btn_tri_i[1]}]
set_property PACKAGE_PIN A10 [get_ports {btn_tri_i[0]}]

create_clock -period 10.000 -name clk100M_in -waveform {0.000 5.000} [get_ports clk100M_in]
create_clock -period 9.091 -name hdmi_rx_clk_p -waveform {0.000 4.546} [get_ports hdmi_rx_clk_p]
create_clock -period 8.000 -name rgmii_rx_clk -waveform {0.000 4.000} [get_ports rgmii_rx_clk]
```

További lépések

- **Implementáció futtatása**
- **FPGA konfigurációs (BIT) fájl generálása**
- **FPGA felkonfigurálása**
  - Hardware Manager, SDK, Vitis
- **Hardver specifikáció exportálása**
  - File menü → Export → Export Hardware…
  - BIT fájllal vagy anélkül
- **Software Development Kit (SDK), Vitis elindítása**
  - File menü → Launch SDK
  - Tools menü → Launch Vitis

Xilinx Software Command-line Tool

- **Lehetőséget biztosít az elkészült hardver alapszintű kipróbálására szoftver nélkül**
  - Memória írás, olvasás → periféria regiszterek elérése
- **Parancssoros eszköz**
- **Kapcsolódás a hardver szerverhez**
  - ***connect*** parancs
- **Cél eszközök listázása, kapcsolódás a cél eszközre**
  - ***targets [sorszám]*** parancs
  - Paraméter megadása nélkül listáz
- **A rendszer alapállapotba állítása**
  - ***rst*** parancs

Xilinx Software Command-line Tool

- **Processzor leállítása és elindítása**
  - ***stop*** és ***con*** parancsok
- **Memória írás**
  - ***mwr [-size b|h|w] <cím> <érték>***
  - ***mwr [-size b|h|w] <cím> {érték1 érték2 …}***
  - Adatméret: 8 bit (b), 16 bit (h), 32 bit (w)
- **Memória olvasás**
  - ***mrd [-size b|h|w] <cím> [olvasások száma]***
  - Adatméret: 8 bit (b), 16 bit (h), 32 bit (w)