

Dr. Kincses Zoltán, Dr. Vörösházi Zsolt:

Beágyazott rendszerek fejlesztése



**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

Beágyazott rendszerek fejlesztése (FPGA)

Dr. Vörösházi Zsolt

5. Periféria modul illesztése (PMOD TMP) hőmérséklet méréssel



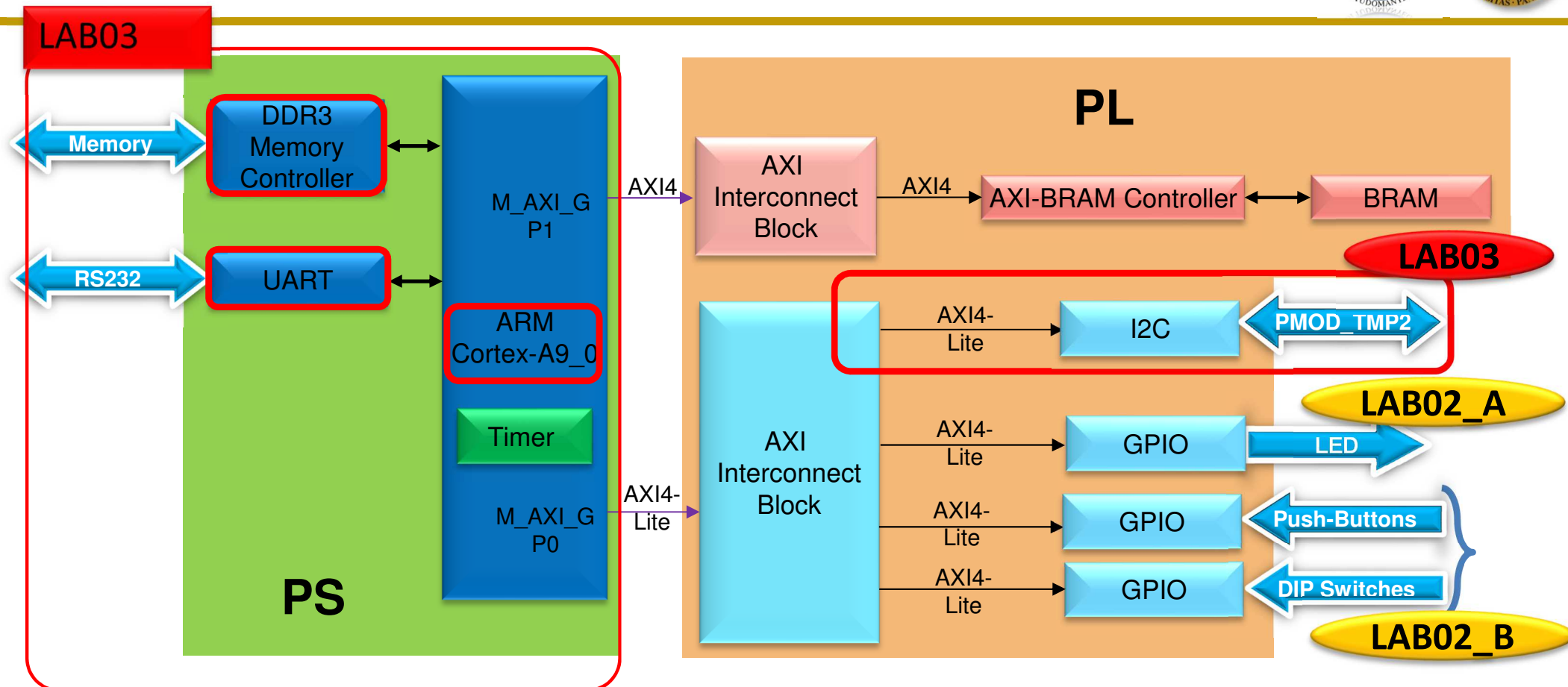
1. Bevezetés – Beágyazott rendszerek, FPGA-k, Digilent ZYBO fejlesztő kártyák és eszközök
2. Beágyazott Rendszer fejlesztő szoftverkörnyezet (Xilinx Vivado Embedded Development) áttekintése
3. Beágyazott alap tesztrendszer (BSB - Base System Builder and Board Bring-Up) összeállítása
4. Perifériák hozzáadása (IP adatbázisból) az összeállított beágyazott alaprendszerhez. Saját periféria hozzáadása az összeállított beágyazott alaprendszerhez
5. **Szoftver alkalmazások fejlesztése, tesztelése: PMOD TMP hőmérséklet mérő**

A feladat megoldásának lépései



- Az előző (04. **fólia**) ismeretkör elsajátítása során létrehozott
 - **File -> Project -> Save As...**
 - Alternatív módszer lehet: projekt archiválása (LAB02B → LAB03 néven), majd pedig a \LAB03 megnyitása Vivado-ban,
- Az IP katalógusból kiválasztott I²C periféria integrálása és összekötése az alaprendszerrel,
- Külső I²C portok létrehozása (.XDC),
- Egy Periféria teszt-alkalmazás (PMOD_Test) készítése az SDK-környezetben,
- A FW+SW tervek teszt verifikálása a **Digilent ZyBo** kártyán.

A bővíthető tesztrendszer



PS oldal:

- ARM hard-processzor mag
- belső OnChip-RAM vezérlő
- RS232 soros interfész
- külső DDR3 memória vezérlő
- (I²C vezérlő opcionális)
- **Timer / interrupt vezérlő (ARM)**

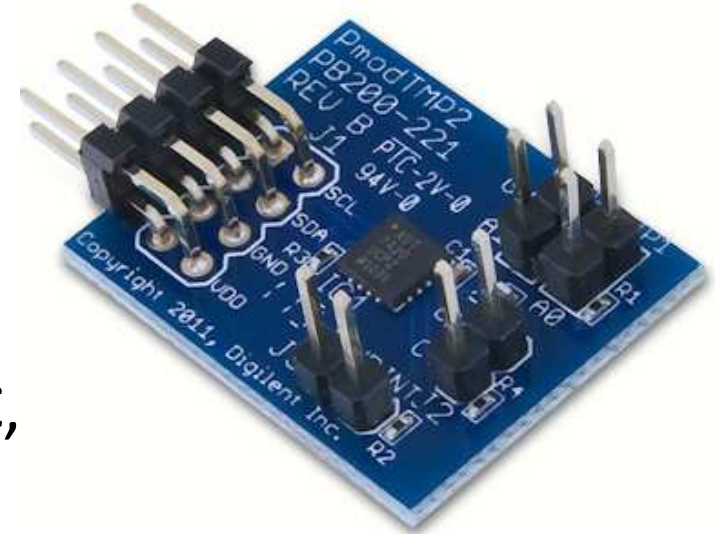
PL oldal:

- **LAB02_A:** GPIO bemenetek
 - PBSs: Push Button (nyomógomb kezelő)
 - DIPs: Switches (kapcsoló kezelő)
- **LAB02_B:** GPIO kimenet
 - LEDs: LED kijelző
- **LAB03: I²C vezérlős hőmérsékletmérő modul (PMOD_TMP2)**
 - **ARM PS/ vagy PL oldalra is csatlakoztatható**

- Digilent PMOD TMP2 hőmérséklet szenzor modul (I²C):
 -  <https://store.digilentinc.com/pmod-tmp2-temperature-sensor/>
 -  <https://reference.digilentinc.com/reference/pmod/pmodtmp2/reference-manual>
- Analog Devices ADT7420 hőmérséklet érzékelő IC:
 -  <http://www.analog.com/media/en/technical-documentation/data-sheets/ADT7420.pdf>
- Analog Devices – Digilent közös Wiki oldala:
 -  <http://wiki.analog.com/resources/alliances/digilent>
- Referencia terv FPGA-ra (SW driver-ekkel)!
 -  <http://wiki.analog.com/resources/fpga/xilinx/pmod/adt7420>
- I²C szabványról általánosan:
 -  Dr. Fodor Attila, Dr. Vörösházi Zsolt: Beágyazott rendszerek, TÁMOP 4.1.2 (PE MIK, Villamosmérnöki és Információs Rendszerek Tanszék) 2011.
<http://tananyagfejlesztes.mik.uni-pannon.hu/>
 -  I2C Wikipedia: <https://en.wikipedia.org/wiki/I%C2%B2C>

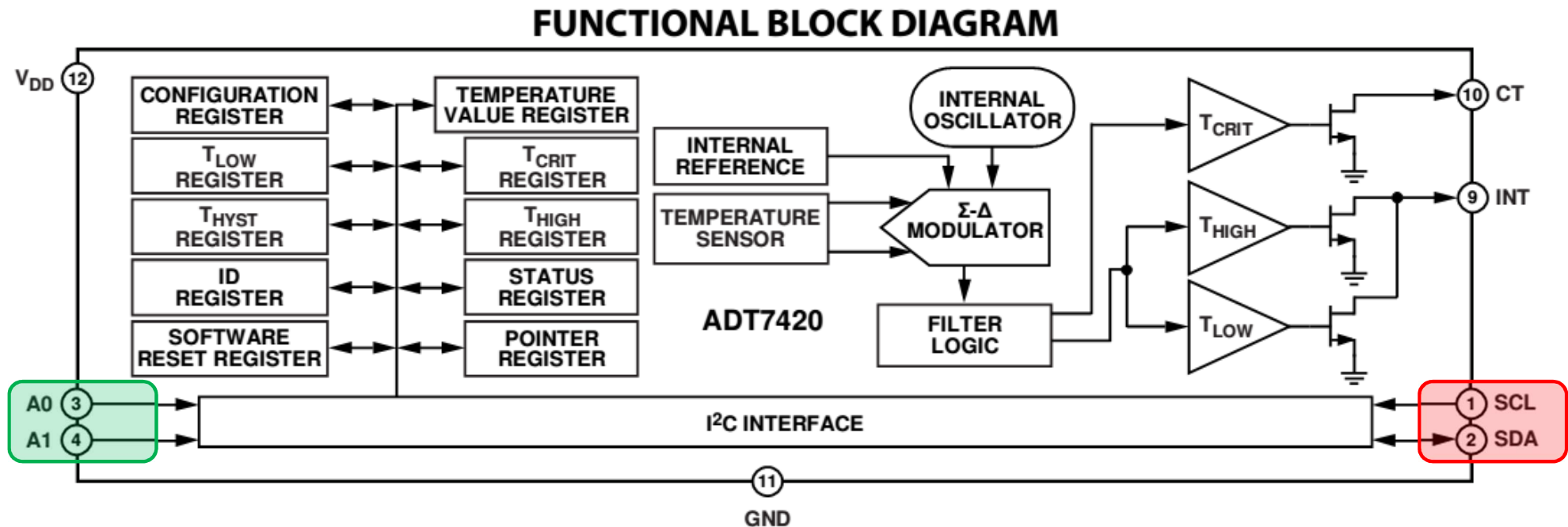
Digilent PMOD_TMP2

- I²C alapú **hőmérséklet érzékelő és hőfokszabályzó periféria modul**
 - $T_A = -40^{\circ}\text{C} \dots +150^{\circ}\text{C}$,
 - max. 16-bit felbontásig skálázható,
 - Átlagos pontossága jobb, mint 0.25°C ,
 - 13 = 9+4 bites módban: $1/2^4 = 0.0625^{\circ}\text{C}$,
 - 16 = 9+7 bites módban: $1/2^7 = 0.0078^{\circ}\text{C}$,
 - **I²C interfész**, 4 választható (jumper) I²C címmel (A_1 - A_0)
 - „Daisy Chain” lehetőség (7/10 bites címezés)
 - Folyamatos konverzió 240ms-ként,
 - Programozható küszöbértékek (max/min - CT), külső lábak, mint threshold (INT),
 - **3.3V** vagy 5V interfész támogatás,
 - Kalibrálást nem igényel!



ADT 7420

- Az áramkör blokk szintű kapcsolási rajza:



- PMOD_TMP2 jelei / választható I²C címek:

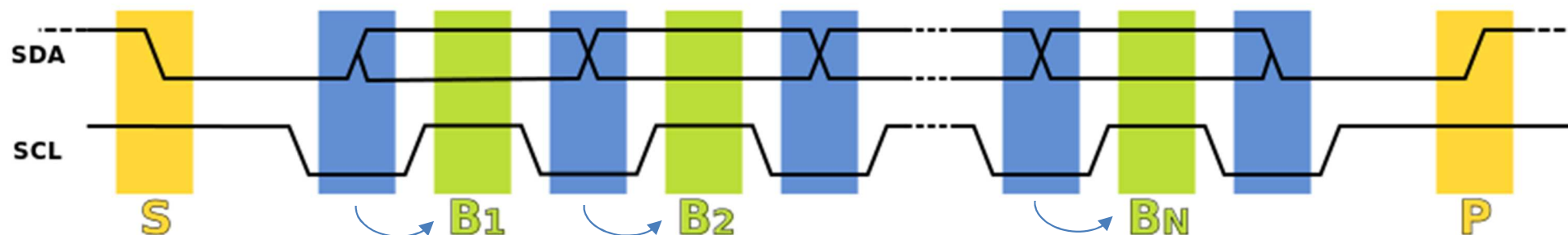
Connector J1 – I2C Communications		
Pin	Signal	Description
1, 2	SCL	I2C Clock
3, 4	SDA	I2C Data
5, 6	GND	Power Supply Ground
7, 8	VCC	Power Supply (3.3V/5V)

Addresses		
JP2	JP1	Address
Open	Open	0x4B (0b1001011)
Open	Shorted	0x4A (0b1001010)
Shorted	Open	0x49 (0b1001001)
Shorted	Shorted	0x48 (0b1001000)

Philips I²C szabványról (1982)



Forrás: <https://en.wikipedia.org/wiki/I%C2%B2C>



1. Data Transfer is initiated with a *START bit* (**S**) signaled by **SDA** being pulled **low** while SCL stays high (pull-down).
2. SDA sets the 1st data bit level while keeping **SCL low** (during **blue** bar time) .
3. The data is sampled (received) when **SCL rises** (**green**) for the first bit (**B1**).
4. This process repeats, SDA transitioning until **SCL is low** again, and the data being read while **SCL is high** (**B2, Bn**).
5. A *STOP bit* (**P**) is signaled when SDA is pulled high while **SCL is high**.

Reserved Address Index	8 Bit Byte			Description
	7 bit Address		R/W Value	
	MSB (4bit)	LSB (3bit)	1 bit	
1	0000	000	0	General Call
2	0000	000	1	START BYTE
3	0000	001	X	CBUS Address
4	0000	010	X	Reserved For Different Bus Format
5	0000	011	X	Reserved For Future Purpose
6	0000	0XX	X	HS-mode Master Code
7	1111	1XX	1	Device ID
8	1111	0XX	X	10-bit slave addressing

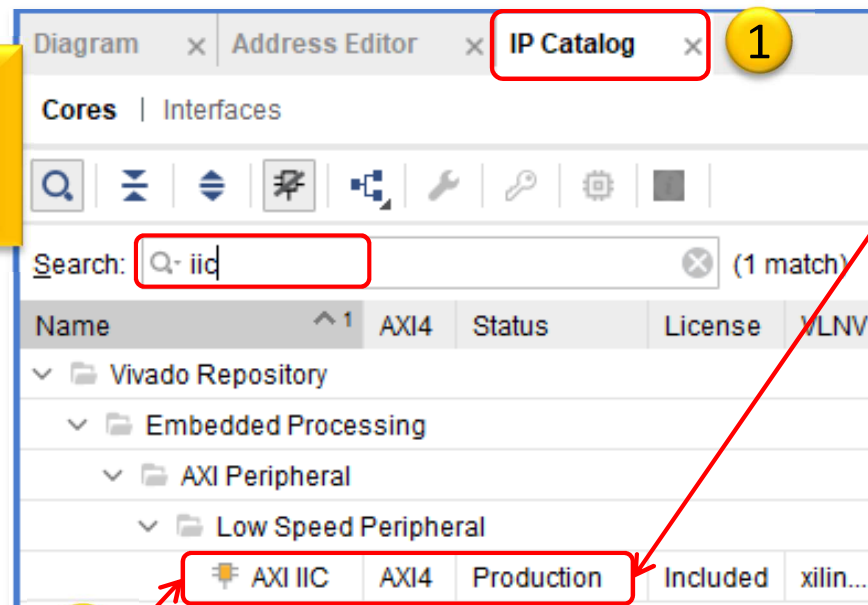
„Master write(0) to / read(1) from the Slave”

I²C vezérlő hozzáadása és összekötése az alaprendszerrel I.



- I²C vezérlő csatlakoztatására két lehetőség van
 - a.) **PL-oldali AXI_I2C IP mag hozzáadása, vagy**
 - b.) PS-oldali PS_I2C0/1 periféria vezérlő engedélyezése Zynq alrendszerben.
- Adjunk a processzor rendszerhez egy AXI_IIC perifériát a Block Diagram segítségével (IP katalógusból)

Váltás, és szűrés az IP katalógus nézetben



IP mag hozzáadása (dupla kattintás)

AXI_IIC IP mag kiválasztása



AXI IIC periféria hozzáadása az alaprendszerhez II.



Dupla kattintással vizsgáljuk meg az AXI_IIC-t:
(minden maradjon alap beállításon)

Customize IP

AXI IIC (2.0)

Documentation IP Location Switch to Defaults

☐ Show disabled ports

Component Name: axi_iic_0

Board IP Configuration 1

IIC Parameters

SCL Clock Frequency (in KHz)	100	[1.0 - 1000.0]
Address mode	7 bit	
SCL Inertial delay (in AXI clocks)	0	[0 - 255]
SDA Inertial delay (in AXI clocks)	0	[0 - 255]
Active state of SDA	1	

Other Parameters

AXI Clock Frequency (in MHz)	25	[25.0 - 300.0]
General Purpose Output width	1	[1 - 8]
Default GPO Port Output Value	0x00	

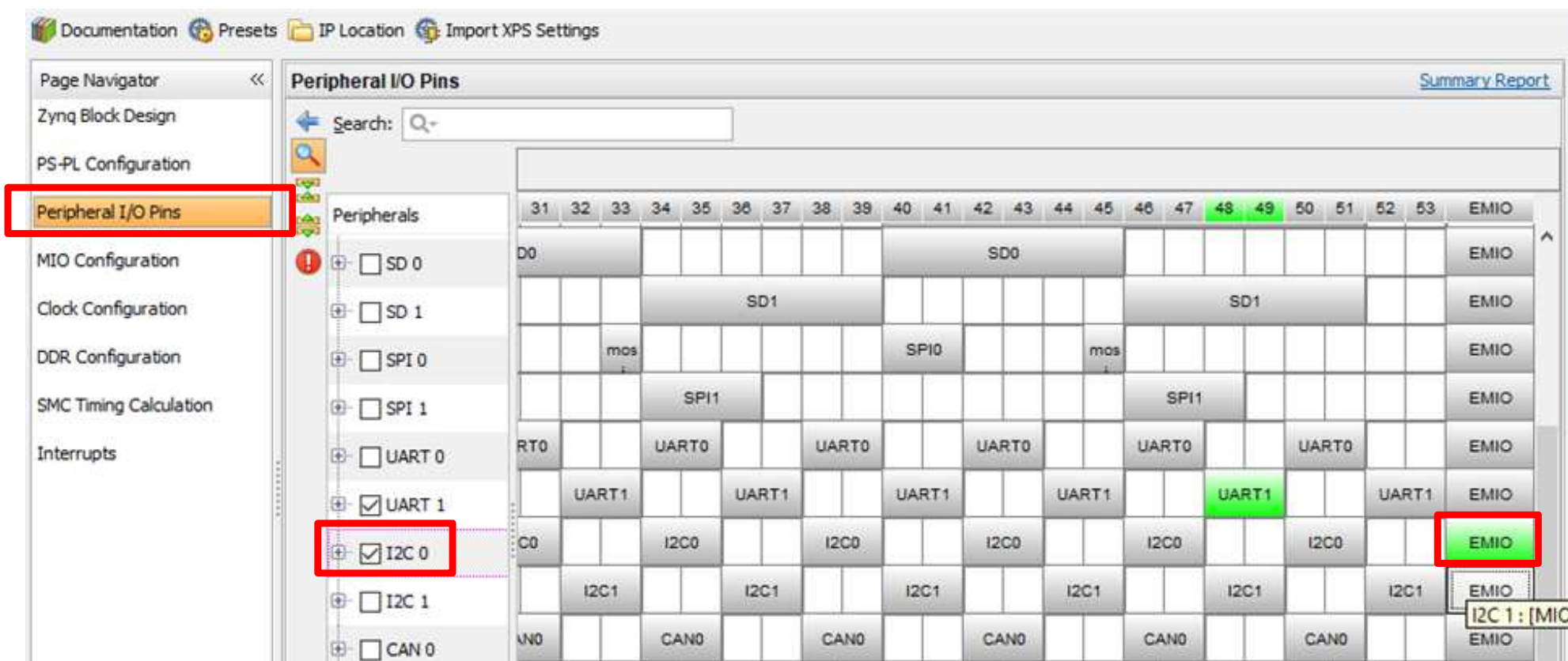
Board interface: marad „custom”.

SCLK: 100 KHz (alap)
Címzés: 7 bites (alap)

GPIO csatona szélessége: 1

Alternatív megoldás: PS I2C

- Nem ezt használjuk, ez csak lehetőség.
- PL (FPGA) oldali I2C blokkot használjuk!



Documentation Presets IP Location Import XPS Settings

Page Navigator << Zynq Block Design PS-PL Configuration **Peripheral I/O Pins** MIO Configuration Clock Configuration DDR Configuration SMC Timing Calculation Interrupts

Peripheral I/O Pins [Summary Report](#)

Search:

Peripherals

- ☐ SD 0
- ☐ SD 1
- ☐ SPI 0
- ☐ SPI 1
- ☐ UART 0
- ☒ UART 1
- ☒ I2C 0
- ☐ I2C 1
- ☐ CAN 0

	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	EMIO					
D0																													EMIO
																													EMIO
																													EMIO
																													EMIO
RT0																													EMIO
																													EMIO
C0																													EMIO
																													EMIO
VN0																													EMIO

I2C 1: MIO

AXI GPIO-k bekötése (autorouter)

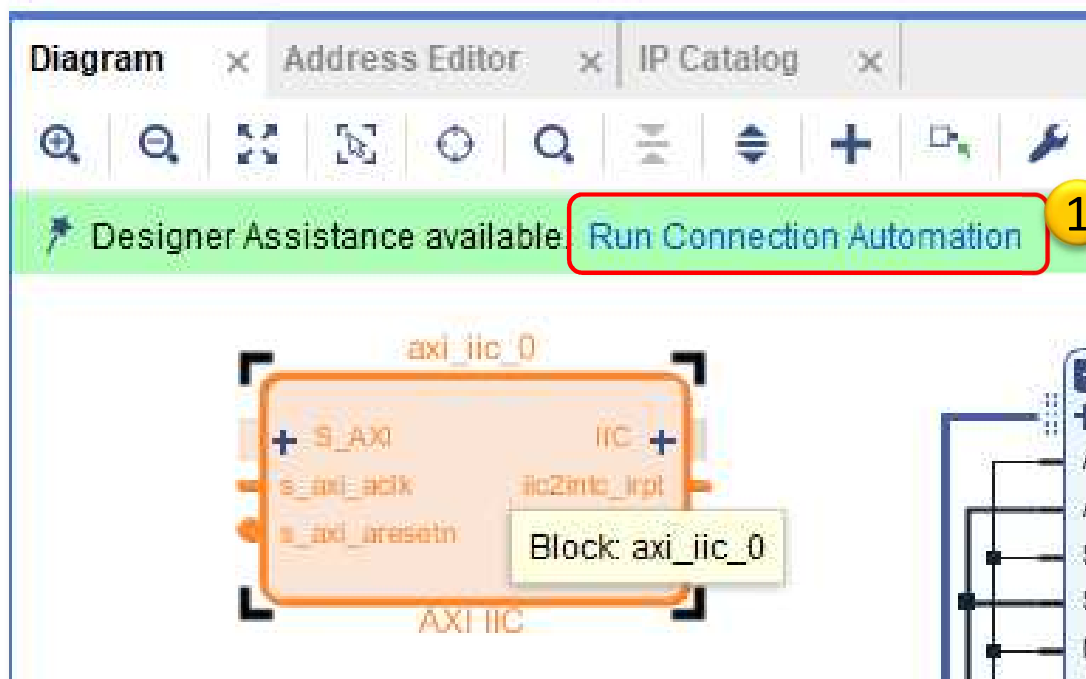
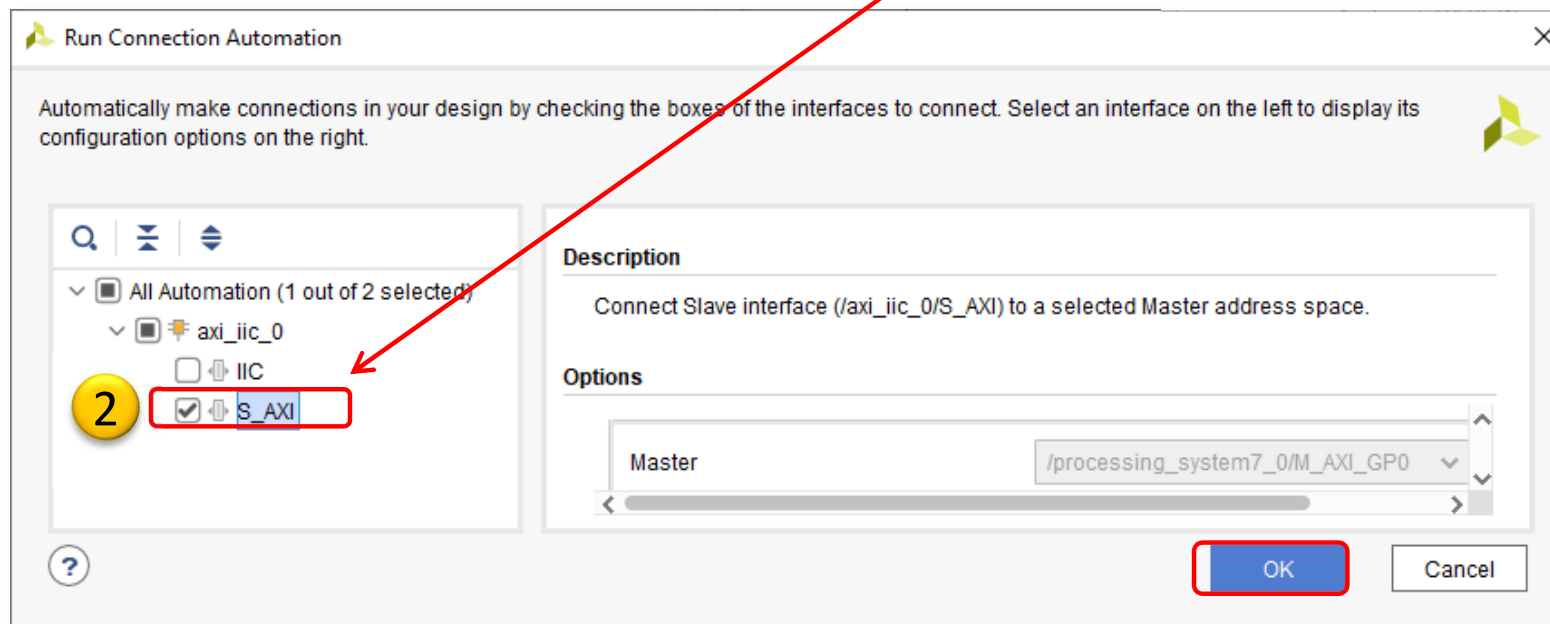


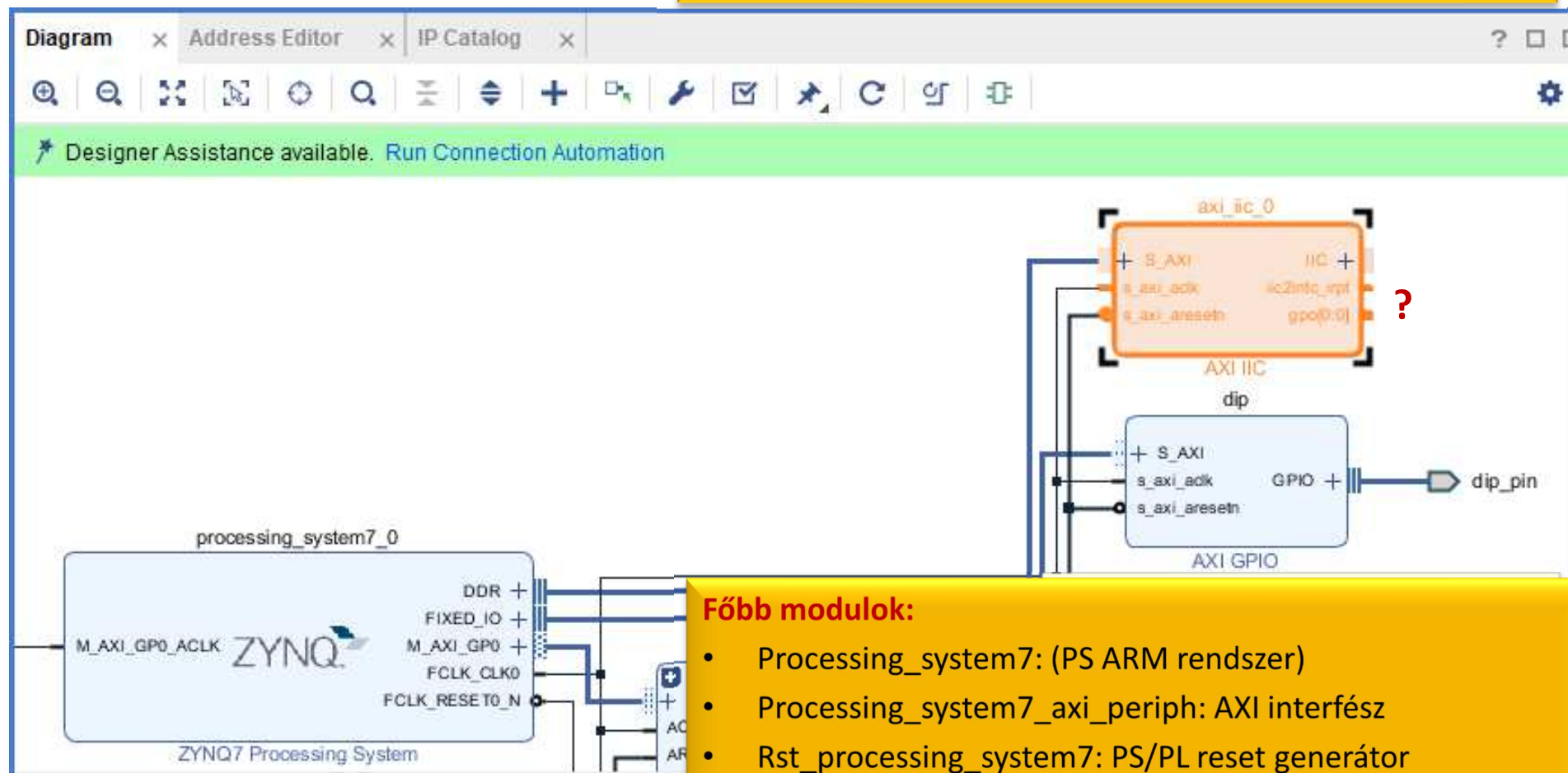
Diagram -> Run
Connection Automation
indítása

AXI_IIC_0 -> S_AXI
interfész kiválasztása,
automata routoláshoz
(minden más
alapértéken marad)



Block design – teljes nézet

Vizsgáljuk meg az egyes blokkokat, illetve összeköttetéseiket!

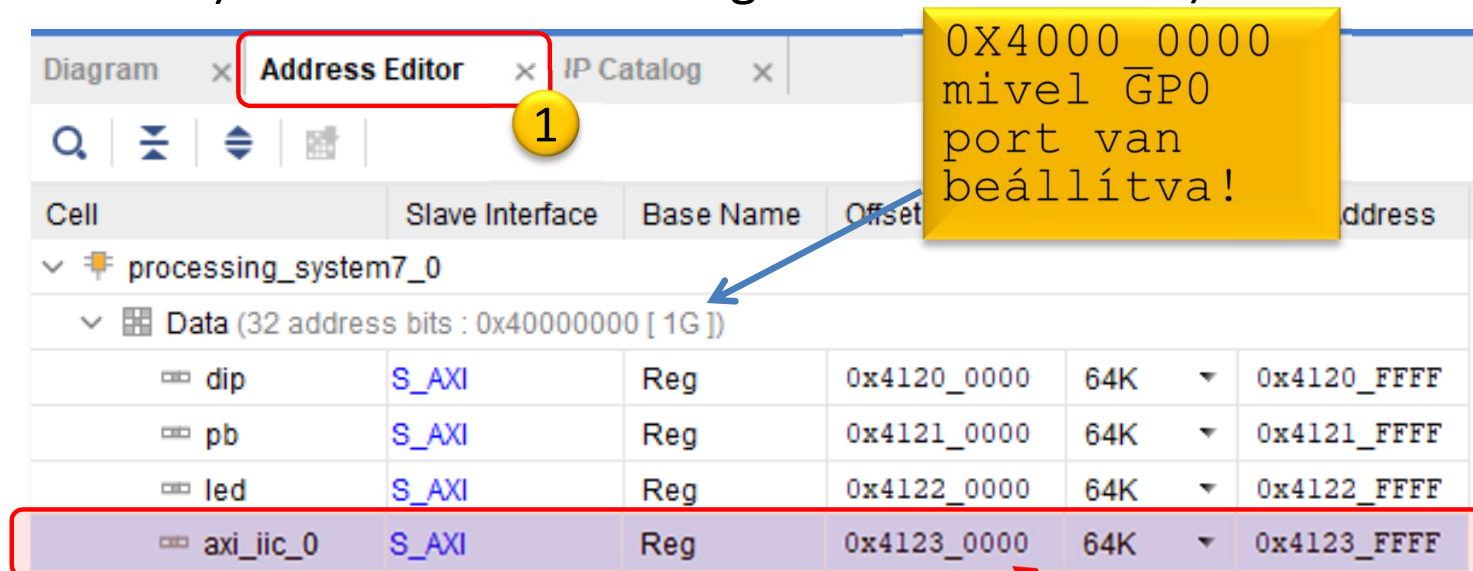


Főbb modulok:

- Processing_system7: (PS ARM rendszer)
- Processing_system7_axi_periph: AXI interfész
- Rst_processing_system7: PS/PL reset generátor
- dip: AXI GPIO a DIP kapcsolóknak (PL oldali)
- Pb: AXI GPIO a PB nyomógomboknak (PL oldali)
- Led: AXI GPIO a LED kijelzőknek (PL oldali)
- Axi_iic_0: AXI_IIC periféria vezérlő (PL oldali)

AXI IIC - Memória címtartományok beállítása

- Block Design -> „Address Editor” nézet kiválasztása
- „Map”-eletlen IP perifériák memória-címtartományhoz rendelése:
 - a.) automatikusan - címgenerálással vs. b.) manuálisan



Cell	Slave Interface	Base Name	Offset	Address
processing_system7_0				
Data (32 address bits : 0x40000000 [1G])				
dip	S_AXI	Reg	0x4120_0000	64K 0x4120_FFFF
pb	S_AXI	Reg	0x4121_0000	64K 0x4121_FFFF
led	S_AXI	Reg	0x4122_0000	64K 0x4122_FFFF
axi_iic_0	S_AXI	Reg	0x4123_0000	64K 0x4123_FFFF

0X4000_0000
mivel GP0
port van
beállítva!

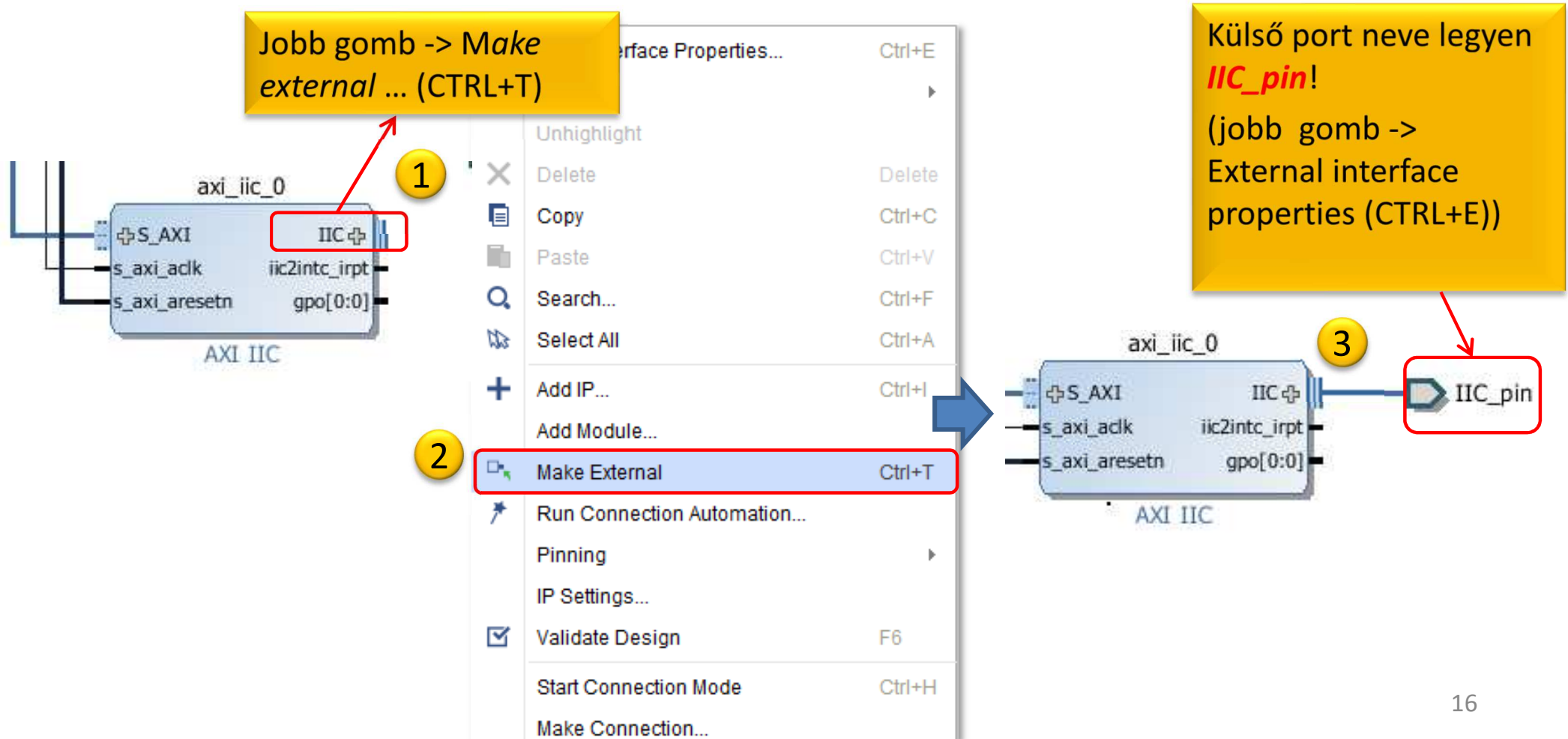
a.) Automatikus
címgenerálás
(jobb gomb ->
Auto Assign
Address)

b.) Base address kézi beállítása*
Axi_iic: 0x4123_0000
(64K)

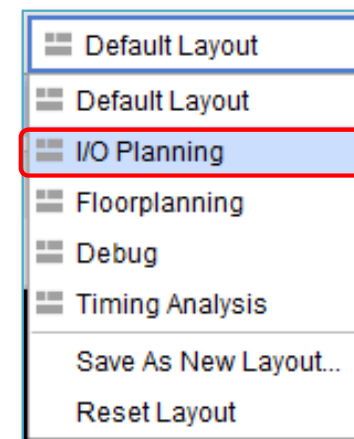
AXI GPIO - Külső portok hozzárendelése

Az AXI_IIC példányhoz hozzá kell még kapcsolni a ZyBo kártyán található (**PMOD JE 3,4 jeleket**) az FPGA (PL-oldali) lábakhoz:

- 1.) Az AXI_IIC port-jait a külső fizikai FPGA lábakra (pin) is kell kötni,
- 2.) Ha kell, a külső portok neveit is definiáljuk (pl: „`_pin`” végződésűre), majd
- 3.) Az <system>.XDC fájl-ban meg kell adni az adott FPGA pin azonosítóját.



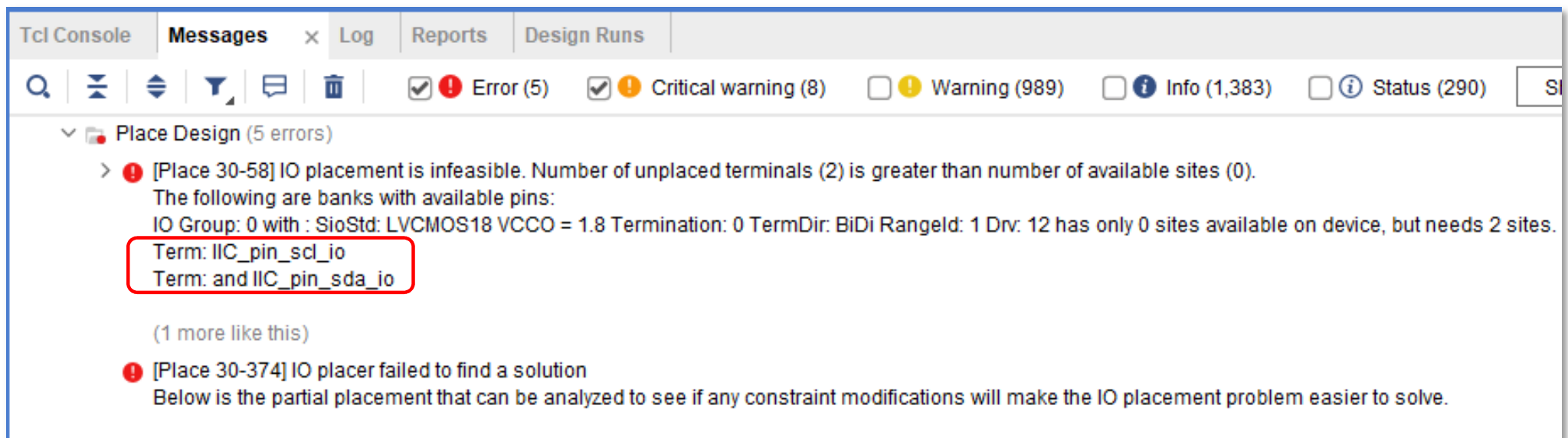
- Ezek után egy-egy a Block Design-t frissíteni kell:
 - Regenerate Layout 
 - Validate Design (DRC) 
 - Flow Navigator -> Run Synthesis 
 - Majd - **Open Synthesized Design** , OK
- Utolsó lépésként a külső porthoz (`iic_pin`) hozzá kell rendelni az FPGA IO lábakat!
 - Layout menü -> I/O Planning layout nézet



IO planning – hibás lábkiosztás



- Amennyiben az előző lépésben nem a szintetizált tervet nyitjuk meg, hanem elindítjuk a „*Run Implementation*”-t akkor az alábbi hibaüzenetet kapjuk:



- Oka: a Vivado megpróbálja elhelyezni a két I²C jelet.
 - Hibás pin location-t és IO Standard-et rendelne (LVCMOS 1.8V) hozzá.

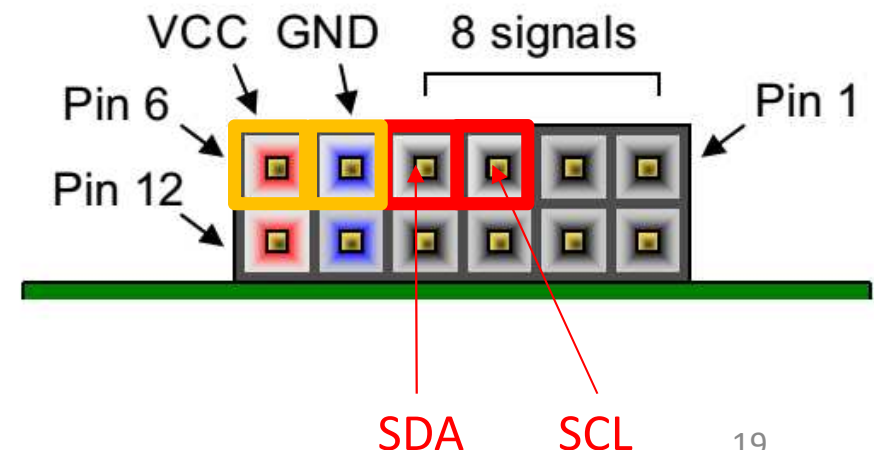
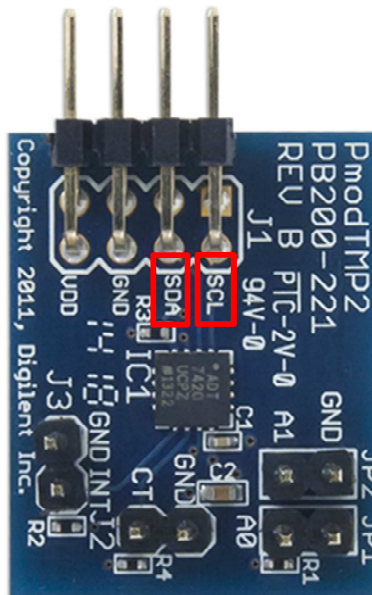
ZyBo - PMOD konnektorok

Pmod JA (XADC)	Pmod JB (Hi-Speed)	Pmod JC (Hi-Speed)	Pmod JD (Hi-Speed)	Pmod JE (Std.)	Pmod JF (MIO)
JA1: N15	JB1: T20	JC1: V15	JD1: T14	JE1: V12	JF1: MIO-13
JA2: L14	JB2: U20	JC2: W15	JD2: T15	JE2: W16	JF2: MIO-10
JA3: K16	JB3: V20	JC3: T11	JD3: P14	JE3: J15	JF3: MIO-11
JA4: K14	JB4: W20	JC4: T10	JD4: R14	JE4: H15	JF4: MIO-12
JA7: N16	JB7: Y18	JC7: W14	JD7: U14	JE7: V13	JF7: MIO-0
JA8: L15	JB8: Y19	JC8: Y14	JD8: U15	JE8: U17	JF8: MIO-9
JA9: J16	JB9: W18	JC9: T12	JD9: V17	JE9: T17	JF9: MIO-14
JA10: J14	JB10: W19	JC10: U12	JD10: V18	JE10: Y17	JF10: MIO-15

Legyen a **JE** jelű standard PMOD konn.

i2c_pin_scl : J15

i2c_pin_sda: H15



IO planning – helyes lábkiosztás I.



- a.) Egyik lehetőség az I/O planning használata (GUI)

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew
IIC_58957 (2)	INOUT			☐		default (LVCMOS18)	1.800		12	SLOW
Scalar ports (2)										
IIC_pin_scl_io	INOUT			☐		default (LVCMOS18)	1.800		12	SLOW
IIC_pin_sda_io	INOUT			☐		default (LVCMOS18)	1.800		12	SLOW

Zybo_master.xdc alapján is megadható lenne a helyes lábkiosztás, melyet a következőkkel kell kiegészíteni:

- Site:
i2c_pin_scl : J15
i2c_pin_sda: H15
- IOSTANDARD: **LVCMOS33**
- Pull type: **PULLUP**
- OCT (On-Chip-Termination): **NONE**

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Termination
IIC_58957 (2)	INOUT			☒	35	LVCMOS33*	3.300		12	SLOW	PULLUP	NONE
Scalar ports (2)												
IIC_pin_scl_io	INOUT		J15	☒	35	LVCMOS33*	3.300		12	SLOW	PULLUP	NONE
IIC_pin_sda_io	INOUT		H15	☒	35	LVCMOS33*	3.300		12	SLOW	PULLUP	NONE

Végül **File -> Save Constraints** vagy **CTRL+S**. Ezután rákérdez az XDC file elmentésére. Adjunk neki nevet: pl. „lab03.xdc”

IO planning – helyes lábkiosztás II.



- b.) Másik lehetőség az .xdc manuális szerkesztése
 - File -> Add Sources... -> Add or Create constraints
 - Create File
 - Adjunk neki nevet: pl. „lab03.xdc”

SYNTHESIZED DESIGN - synth_1 | xc7z010clg400-1 (active)

Sources | Netlist | Device Constraints

Design Sources (1)
design_1_wrapper(STRUCTURE) (design_1_wrapper.vhd)
design_1_i: design_1 (design_1.bd) (1)

Constraints (1)
constrs_1 (1)
lab03.xdc (target)

Simulation Sources (1)
Utility Sources

Package | Device | design_1_wrapper.vhd | **lab03.xdc**

E:/BERSZT_2019/lab_03/lab_03.srscs/constrs_1/new/lab03.xdc

```
1 set_property OFFCHIP_TERM NONE [get_ports IIC_pin_scl_io]
2 set_property OFFCHIP_TERM NONE [get_ports IIC_pin_sda_io]
3 set_property PULLUP true [get_ports IIC_pin_scl_io]
4 set_property PULLUP true [get_ports IIC_pin_sda_io]
5 set_property PACKAGE_PIN J15 [get_ports IIC_pin_scl_io]
6 set_property PACKAGE_PIN H15 [get_ports IIC_pin_sda_io]
7 set_property IOSTANDARD LVCMOS33 [get_ports IIC_pin_scl_io]
8 set_property IOSTANDARD LVCMOS33 [get_ports IIC_pin_sda_io]
9
```

Végül **File -> Save Constraints** vagy **CTRL+S**. Ezután rákérdez az XDC file elmentésére.

Pmod JE (Std.)
JE1: V12
JE2: W16
JE3: J15
JE4: H15

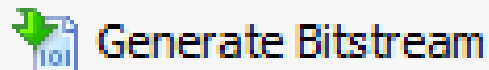
- Flow Navigator menü → Run Implementation



- Kiszűri az esetleges elkötéseket, hibákat,
- Figyelmeztető (warning) jellegű üzenetek megengedettek (implementálható a terv),
- Legtöbb lebegő (floating) vezetékekkel sem kell foglalkozni (pl. Peripheral Reset, stb).
- **Miközben dolgozik – hosszabb idő - érdemes megnézni a fordítási riportokat!**

Ezután indítható el a Bitstream generálás.

- Flow Navigator → Generate Bitstream.



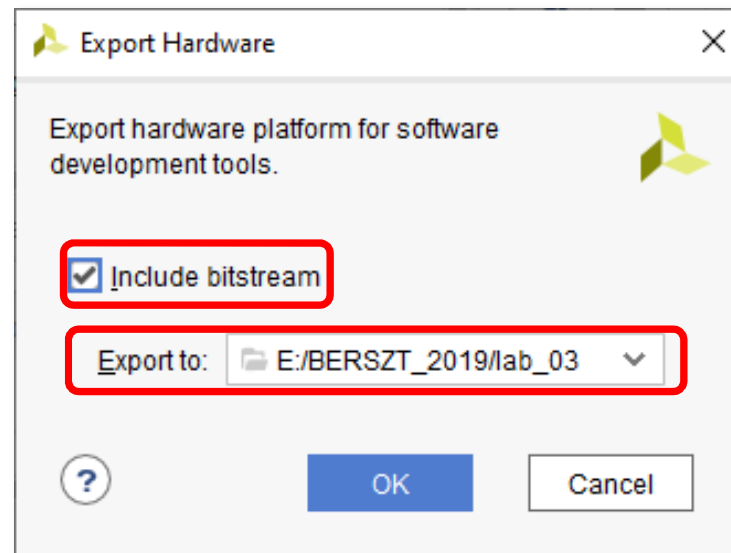
Xilinx Vivado SDK

SZOFTVER TESZT ALKALMAZÁS ÖSSZEÁLLÍTÁSA (PMOD_TMP2)

- 
1. Vivado projekt létrehozása, majd Export → **SDK**,
 2. **Új**, vagy C/C++ template-ből generálható alkalmazás létrehozása (TestApp PMOD alkalmazás):
 - a) **BSP** (Board Support Package) generálása és fordítása,
 - b) **Linker Script** generálása (memória szekciók megadása, .ld),
 - c) **SW** alkalmazáskód megírása/generálása, és fordítása. (**zip!**)
 3. Soros terminál beállítása (USB-soros port beállítása),
 4. JTAG-USB programozó csatlakoztatása és beállítása,
 5. *FPGA konfigurálása!! (.bit, mivel PL oldal is használt)*
 6. *,debug' – hardveres hibakeresés beállítása*
 7. Debug (breakpoint-ok beszúrása, léptetés, stb.)

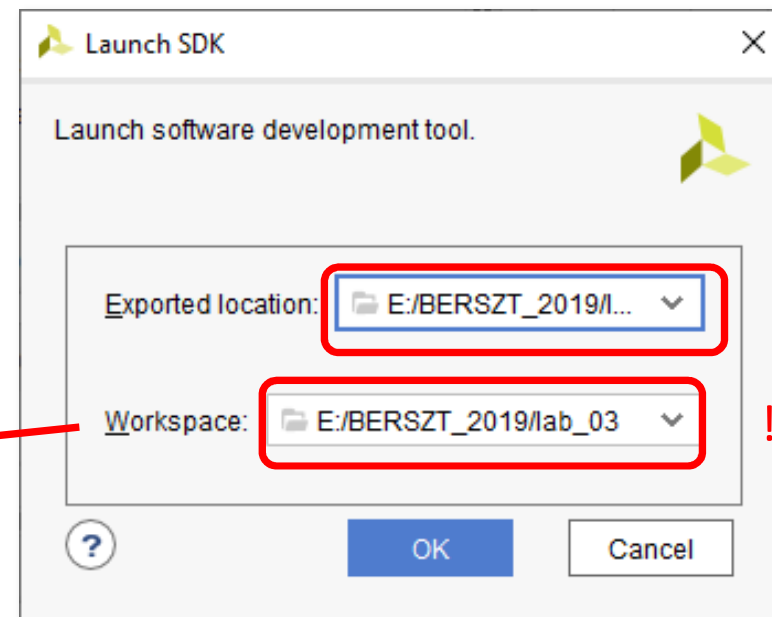
Export HW -> Vivado SDK

- File -> Export
-> Export
hardware



Mivel PL (FPGA) oldal is konfigurálva lett, ezért most már kell a bitstream is az SDK-hoz.
Choose Location... **lab03** kiválasztása!

- File -> Launch
SDK, OK.
 - Vivado SDK indítása a keretrendszerből (ezt kívülről indítva is meg lehetne tenni)

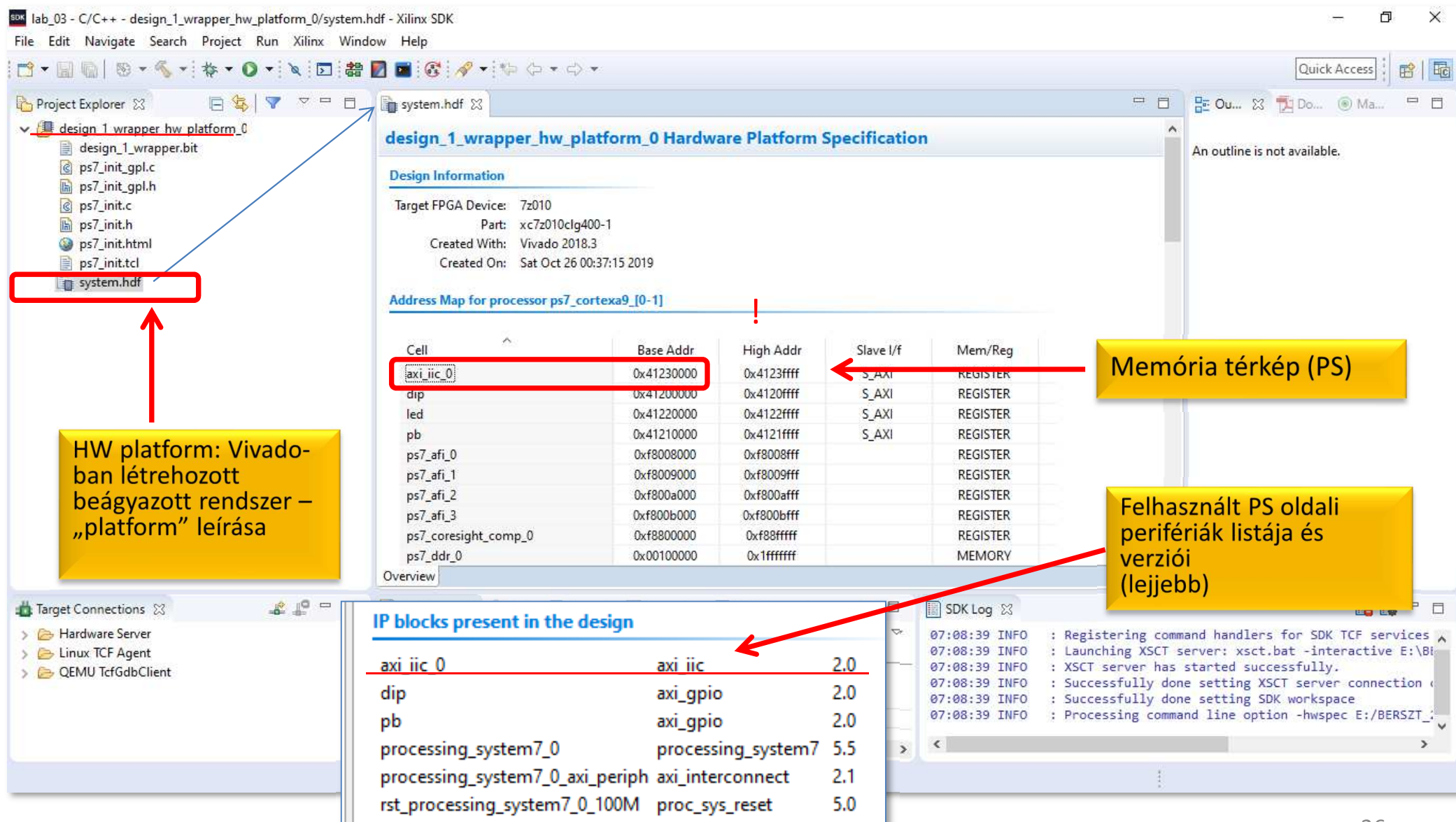


Fontos, hogy a workspace-t lehetőleg ne az előző projektbe hozza létre! -> *Choose location...* – és mostani **lab03** kiválasztása

Vivado SDK GUI

.hdf = Hardware description file = hardver konfigurációs csomag file.

(Valójában egy zip, amely két leíró xml-t tartalmaz!) Ebből olvassa ki az SDK a beállításokat.



The screenshot shows the Vivado SDK GUI with the following components:

- Project Explorer:** Shows the project structure for 'design_1_wrapper_hw_platform_0'. The file 'system.hdf' is highlighted with a red box and a red arrow pointing to it.
- Hardware Platform Specification:** Displays the design information for 'design_1_wrapper_hw_platform_0'. The 'Address Map for processor ps7_cortexa9_0-1' table is shown, with the 'axi_iic_0' entry highlighted by a red box and a red arrow pointing to it.
- IP blocks present in the design:** A table listing the IP blocks and their versions, with the 'axi_iic_0' entry highlighted by a red box and a red arrow pointing to it.
- SDK Log:** Shows the log of the SDK process, including the registration of command handlers and the successful launch of the XSCT server.

Annotations in the image:

- HW platform: Vivado-ban létrehozott beágyazott rendszer – „platform” leírása** (Yellow box pointing to 'system.hdf')
- Memória térkép (PS)** (Yellow box pointing to the 'Address Map' table)
- Felhasznált PS oldali perifériák listája és verziói (lejjebb)** (Yellow box pointing to the 'IP blocks present in the design' table)

Cell	Base Addr	High Addr	Slave I/f	Mem/Reg
axi_iic_0	0x41230000	0x4123ffff	S_AXI	REGISTER
dip	0x41200000	0x4120ffff	S_AXI	REGISTER
led	0x41220000	0x4122ffff	S_AXI	REGISTER
pb	0x41210000	0x4121ffff	S_AXI	REGISTER
ps7_afi_0	0xf8008000	0xf8008fff		REGISTER
ps7_afi_1	0xf8009000	0xf8009fff		REGISTER
ps7_afi_2	0xf800a000	0xf800afff		REGISTER
ps7_afi_3	0xf800b000	0xf800bfff		REGISTER
ps7_coresight_comp_0	0xf8800000	0xf880ffff		REGISTER
ps7_ddr_0	0x00100000	0x1fffffff		MEMORY

Block Name	IP Name	Version
axi_iic_0	axi_iic	2.0
dip	axi_gpio	2.0
pb	axi_gpio	2.0
processing_system7_0	processing_system7	5.5
processing_system7_0_axi_periph	axi_interconnect	2.1
rst_processing_system7_0_100M	proc_sys_reset	5.0

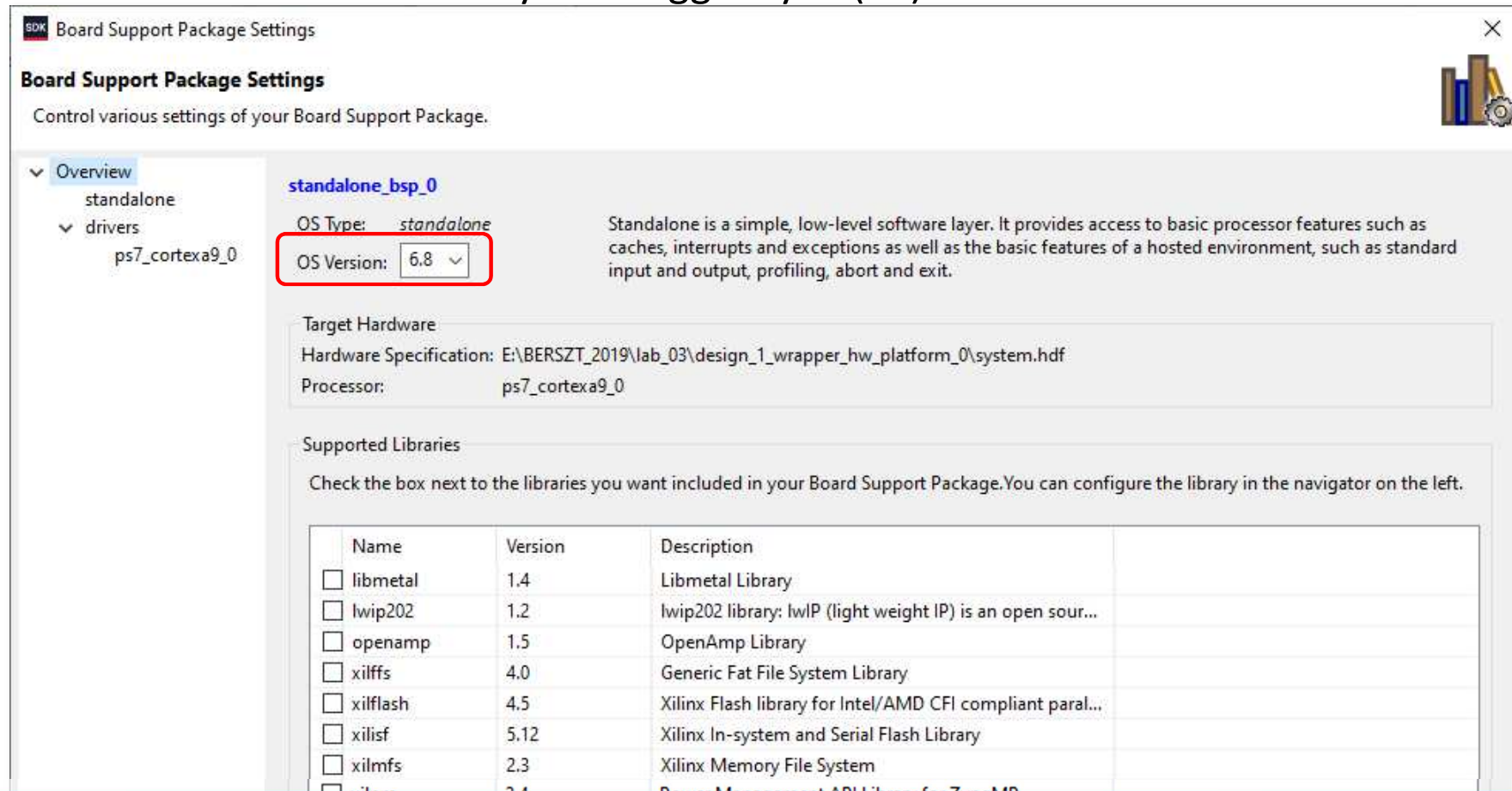
- 1.) A Launch SDK - **Choose Location...** opciójával válasszuk ki a `<project_dir>/LAB03/` → majd kattintsunk az **OK**-ra.
- 2.) Az SDK-ban: **File** → **New** → **Board Support Package** → kattintsunk a **Finish**-re.
 - Új BSP neve `standalone_bsp_0` is maradhat,
 - Hagyjunk mindent alap értéken.

Tesztalkalmazás (TestApp) készítése II./a



- **Software Platform Settings**

- Operációs rendszer kiválasztása: *standalone* vs. *xilkernel* (esetleg 3rd Party OS)
- Rendelkezésre álló könyvtári függvények (lib) kiválasztása



OK -> Legenerálódik az **xparameters.h** és további mappák.

Helye: <projectdir>/lab03/standalone_bsp_0/ps7_cortexa9_0/include mappa



Tesztalkalmazás (TestApp) készítése II./b

standalone_bsp_0 Board Support Package

Modify this BSP's Settings Re-generate BSP Sources

Board Support Package Settings

Control various settings of your Board Support Package.

Overview
standalone
drivers
ps7_cortexa9_0

Drivers

The table below lists all the components found in your hardware system. You can modify the driver (or its version) assigned for each component. If you do not want to assign a driver to a component or peripheral, please choose 'none'.

Component	Component Type	Driver	Driver Version
ps7_cortexa9_0	ps7_cortexa9	cpu_cortexa9	2.7
axi_iic_0	axi_iic	iic	3.5
dip	axi_gpio	gpio	4.3
led	axi_gpio	gpio	4.3
pb	axi_gpio	gpio	4.3
ps7_afi_0	ps7_afi	generic	2.0
ps7_afi_1	ps7_afi	generic	2.0

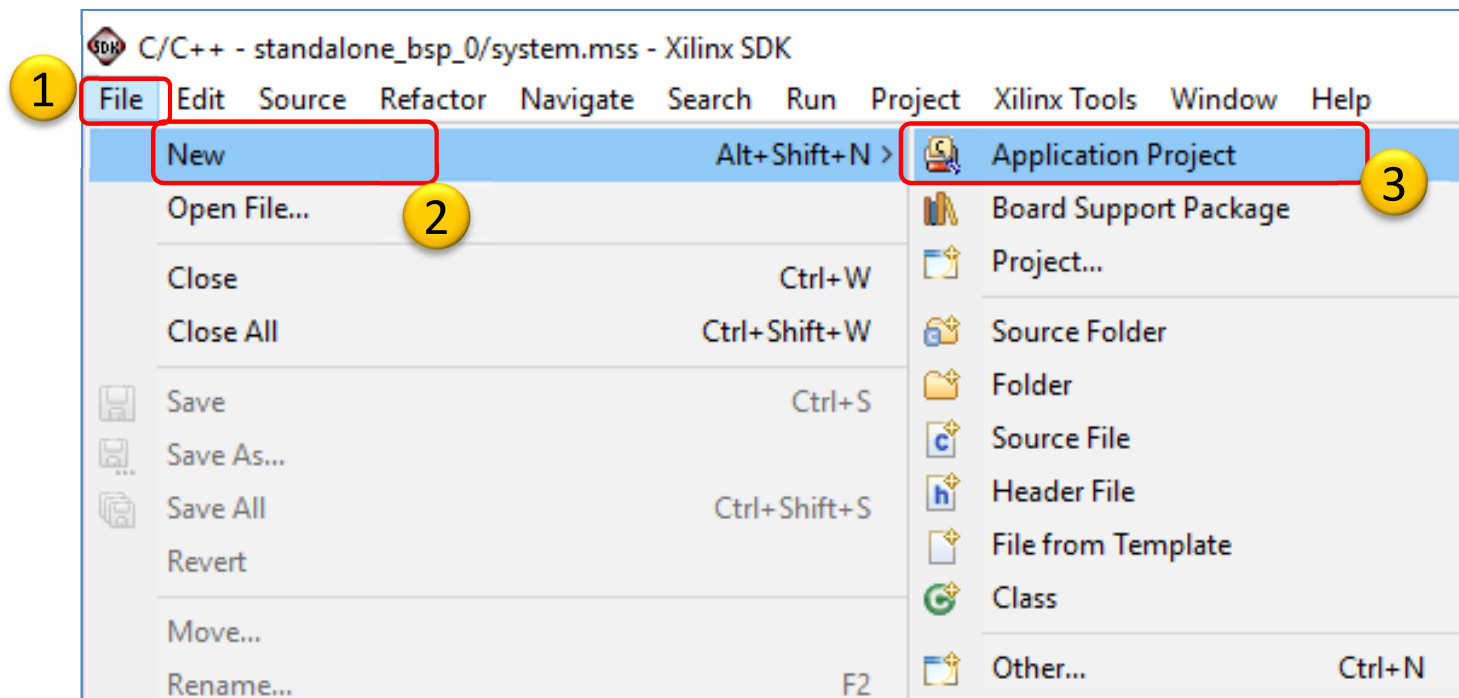
OK Cancel

Fontos: axi_iic_0 perifériához a none/generic driver helyett az „iic” drivert kell kiválasztani a helyes működéshez (ha nem ez lenne!)

Tesztalkalmazás (TestApp) készítése III.



- Új projekt alkalmazás létrehozása
(File → New → Application Project)



BSP: *standalone_bsp_0*
(alapértelmezett)

Tesztalkalmazás (TestApp) készítése IV.



SDK New Project

Application Project
Create a managed make application project.

1
Project name: **TestApp**

☒ Use default location
Location: E:\BERSZT_2019\lab_03\TestApp
Choose file system: default

OS Platform: standalone

Target Hardware
Hardware Platform: design_1_wrapper_hw_platform_0
Processor: ps7_cortexa9_0

Target Software
Language: ☒ C ☐ C++
Compiler: 32-bit
Hypervisor Guest: N/A
Board Support Package: ☐ Create New ☒ Use existing standalone_bsp_0

2
TestApp alkalmazás már létező,
korábban létrehozott BSP-hez
rendelése (*standalone_bsp*)

< Back Next > Finish Cancel

Projekt neve: *TestApp*

SDK New Project

Templates
Create one of the available templates to generate a fully-functioning application project.

Available Templates:

- Dhrystone
- Empty Application**
- Hello World
- IwIP Echo Server
- IwIP TCP Perf Client
- IwIP TCP Perf Server
- IwIP UDP Perf Client
- IwIP UDP Perf Server
- Memory Tests
- OpenAMP echo-test
- OpenAMP matrix multiplication Demo
- OpenAMP RPC Demo
- Peripheral Tests
- RSA Authentication App
- Zynq DRAM tests
- Zynq FSBL

3

A blank C project.

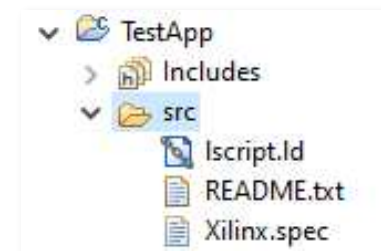
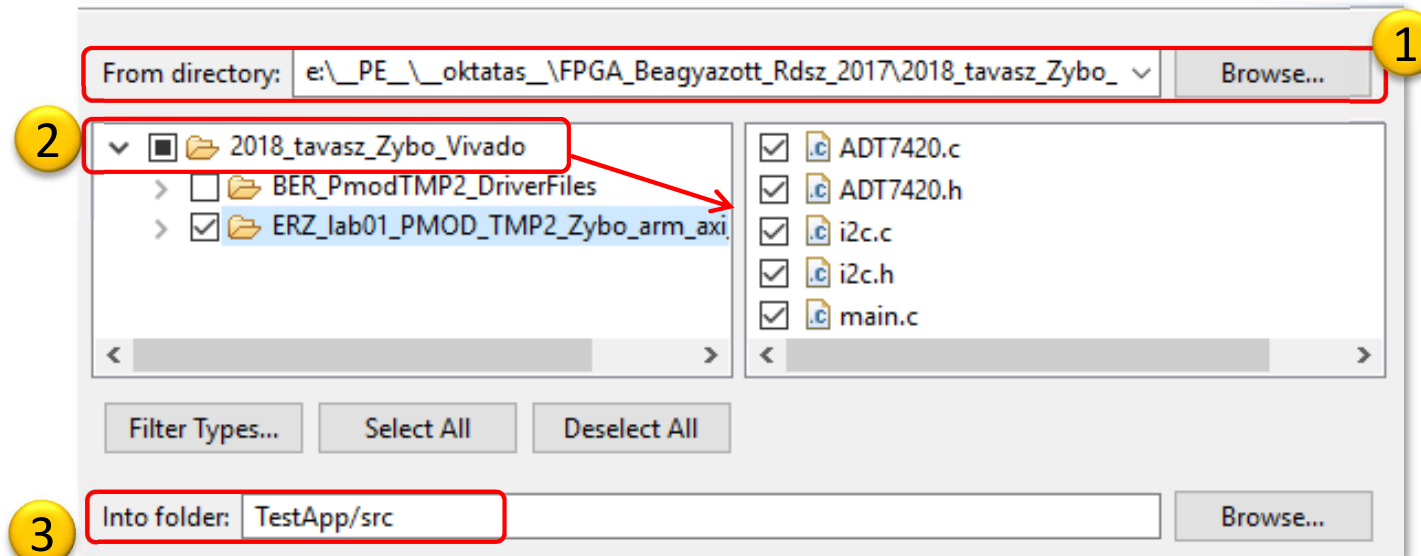
? < Back Next > **Finish** Cancel

Tesztalkalmazás (TestApp) készítése V.



Forrás fájl(ok) importálása: TestApp/src kijelölése!

- File -> Import ... -> General -> Archive File-> NEXT!



BER_PmodTMP2_DriverFiles.zip * -ből hozzáadni:

- ADT7420.c / .h
- I2c.c / .h
- Main.c

* Tárgy oldaláról letölthető: [BER_PmodTMP2_DriverFiles.zip](#)

- Linker script generálása (.ld): OnChip RAM vs. külső MEM.
- SW alkalmazás fordítása (.elf)
- Tesztverifikáció: bitfile generálás + tesztelés FPGA-n.

*Megjegyzés: az i2c.c és i2.h fájlok az Analog Devices támogató csomagjából lettek felhasználva. Alternatív megoldásként a Xilinx iic.c és iic.h fájlokat is használhatjuk:

Lásd: <Xilinx install_dir>/.../sw/XilinxProcessorIPLib/drivers/iic_vX_YZ_a/doc/html/api/index.html

TestApp forráskód hibáinak javítása



- 1.)

```
In file included from ../src/ADT7420.c:49:0:
../src/ADT7420.c: In function 'ADT7420_Init':
../src/ADT7420.h:49:22: error: 'XPAR_AXI_IIC_BASEADDR' undeclared (first use in this function)
#define IIC_BASEADDR XPAR_AXI_IIC_BASEADDR
                        ^|
```

 - Megoldás: adjuk meg az XPAR_AXI_IIC_BASEADDR #define helyes nevét az `xparameters.h` file alapján
- 2.)

```
../src/ADT7420.h:53:28: error: expected expression before '<' token
#define ADT7420_IIC_ADDR    <address>
                        ^
```

 - Megoldás: nézzük meg a ADT7420_IIC_ADDR helyes címét (korábbi fólia, v. kártya adatlap)
- 3.)

```
../src/main.c: In function 'main':
../src/main.c:78:12: error: 'XPAR_RS232_UART_1_BASEADDR' undeclared (first use in this function)
Xil_Out32(XPAR_RS232_UART_1_BASEADDR+0x0C, (1 << 4));
```

 - Megoldás: keressünk ilyen nevű define-t az `ADT7240.h` file-ban.

Kérdés:

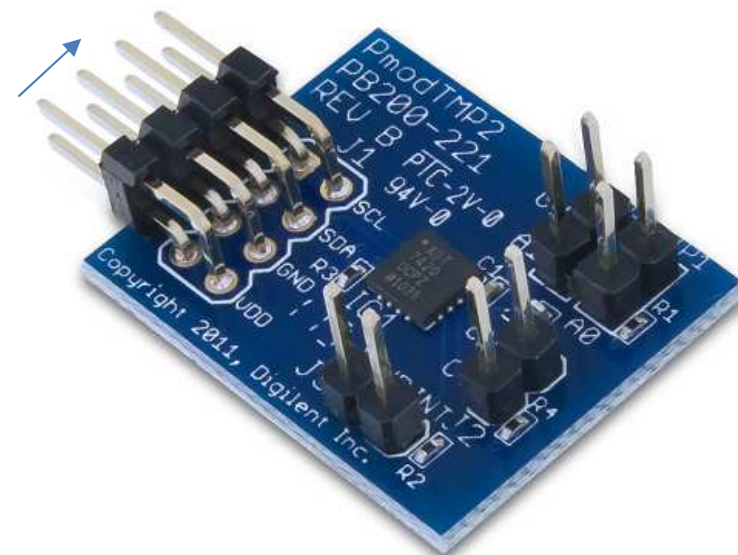
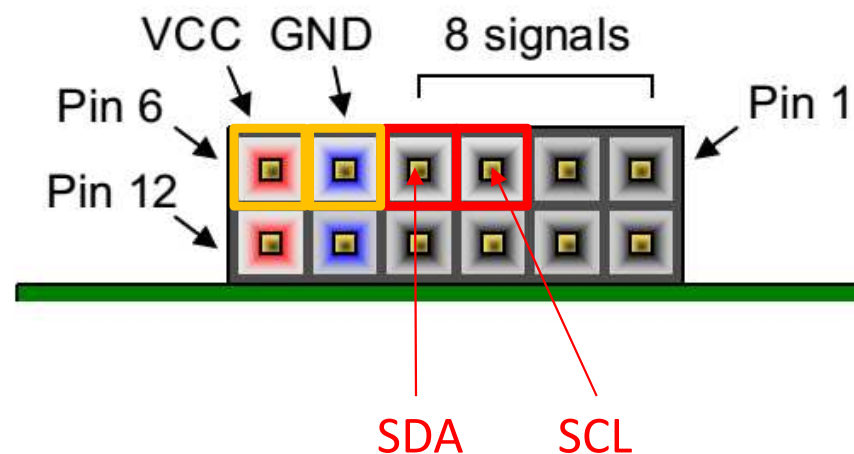
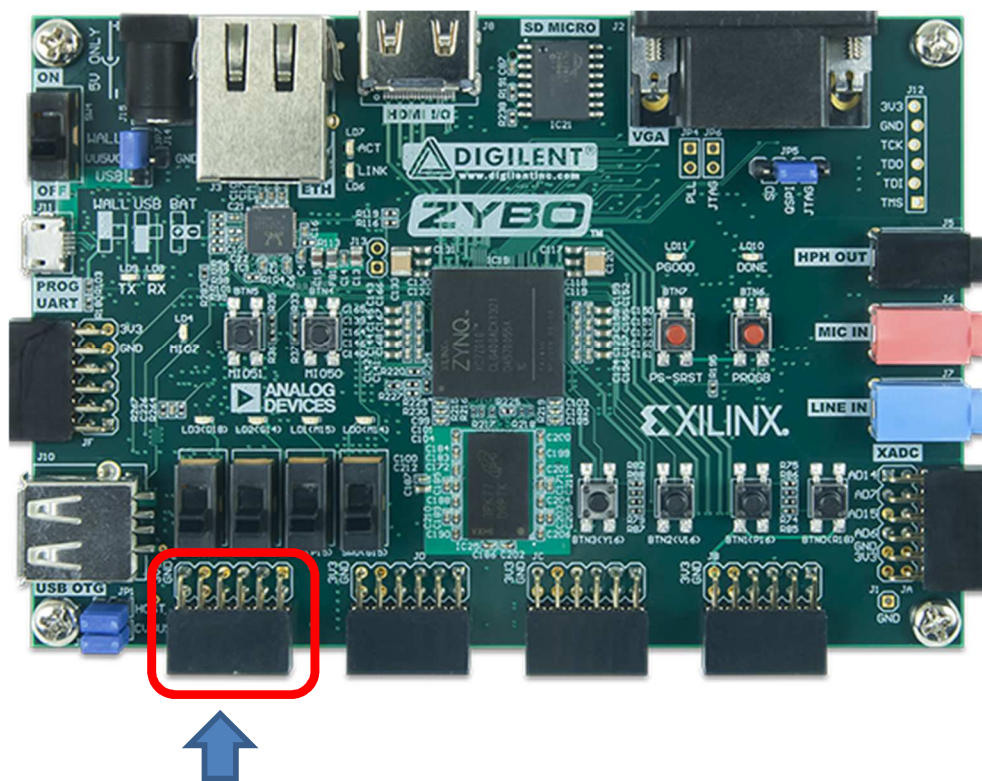


- Mekkora a letöltendő .elf file mérete?
- Megoldás:

```
'Invoking: ARM Print Size'  
arm-xilinx-eabi-size TestApp.elf |tee  
"TestApp.elf.size"  
    text    data    bss    dec    hexfilename  
    27780    1160    22580    51520    c940TestApp.elf  
'Finished building: TestApp.elf.size'
```

Teszt verifikáció I.

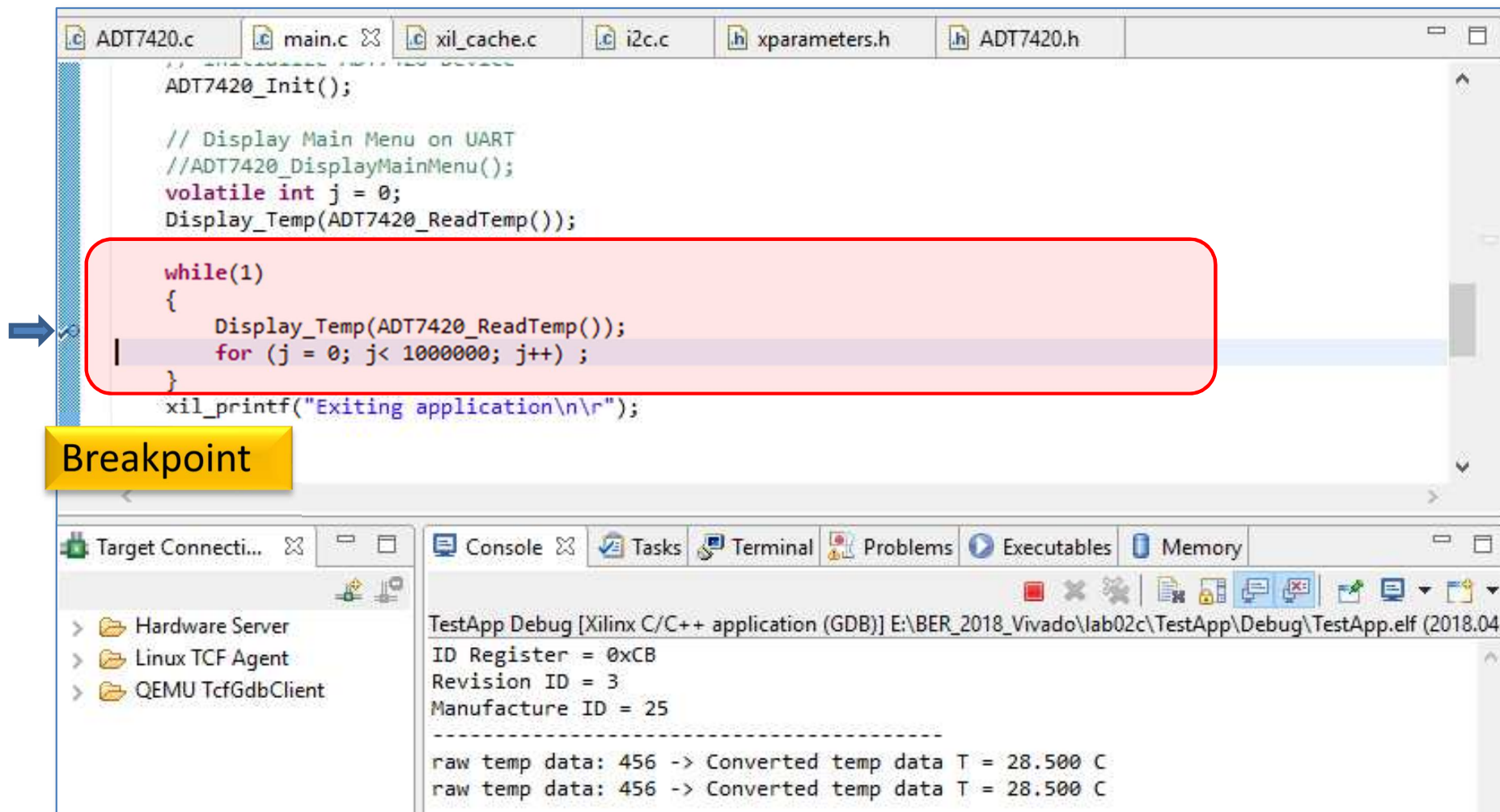
- Kártya helyes csatlakoztatása PMOD **JE**-re.



- 1.) Digilent PMOD_TMP2 hőmérséklet mérő kártya megfelelő illesztése a Digilent ZyBo kártyához:
 - **Standard JE** PMOD foglalat felső sorának jobbról 3-4-es lábaira (SCL – SDA)
 - PMOD_TMP2-es kártyán a JP1/JP2 jumperek rövidre zártak? (A0/A1 címek kiosztása ennek megfelelően !!)
- 2.) A JTAG-USB programozó/táp (USB-soros port) csatlakoztatása a kártyához és a számítógéphez,
- 3.) Digilent ZyBo kártya bekapcsolása,
- **4.) Xilinx Tools → Program FPGA**
 - .bit fájl kiválasztása,
 - FPGA programozása,
- **5.) Debug konfiguráció: környezet beállítása és indítása:**
 - Consol-ra az USB soros port beállítása (COMx),
 - Debug config-ban megadható akár az FPGA .bit file konfigurációja is (Entire reset).

Teszt verifikáció III.

- A forráskódból a következő rész felel az aktuális hőmérséklet kiíratásáért:
 - `Display_Temp(ADT7420_ReadTemp());`
 - Javaslat: tegyük a függvényt végtelen ciklusba a folyamatos kiíratáshoz.



```
ADT7420.c | main.c | xil_cache.c | i2c.c | xparameters.h | ADT7420.h
ADT7420_Init();

// Display Main Menu on UART
//ADT7420_DisplayMainMenu();
volatile int j = 0;
Display_Temp(ADT7420_ReadTemp());

while(1)
{
    Display_Temp(ADT7420_ReadTemp());
    for (j = 0; j< 1000000; j++) ;
}

xil_printf("Exiting application\n\r");
```

Breakpoint

Target Connecti... | Console | Tasks | Terminal | Problems | Executables | Memory

TestApp Debug [Xilinx C/C++ application (GDB)] E:\BER_2018_Vivado\lab02c\TestApp\Debug\TestApp.elf (2018.04.)

ID Register = 0xCB
Revision ID = 3
Manufacture ID = 25

raw temp data: 456 -> Converted temp data T = 28.500 C
raw temp data: 456 -> Converted temp data T = 28.500 C

Eredmény (debug log)

- Készítsük el annak Debug Configuration-ját, majd Debug-oljuk az alkalmazást.

ID Register = 0xCB

Revision ID = 3

Manufacture ID = 25

raw temp data: 456 -> Converted temp data T = 28.500 C

raw temp data: 456 -> Converted temp data T = 28.500 C

...