FPGA fejlesztés a Xilinx ISE Webpack-ben

Írta: Varga László Szerkesztette: Molnár Zsolt

BMF KVK MAI

2007. október 25.

Tartalomjegyzék

1. Bevezetés	3
2. Mintafeladat megoldása – kapcsolási rajz alapú tervezés	5
3. Feladatok a kapcsolási rajz alapú tervezéshez	22
4. Mintafeladat kiegészítése állapotdiagram modellel	23
5. Feladatok az állapotdiagram alapú tervezéshez	28
6. VHDL modul fejlesztése	29
7. Feladatok a VHDL alapú tervezéshez	34
1. sz. melléklet – Az általunk használt gyakorlópanel lábkiosztása	35
2. sz. melléklet – Néhány egyszerű VHDL példa	36

1. Bevezetés

A Xilinx ISE WebPack (Integrated Software Environment – integrált szoftverkörnyezet) a Xilinx cég FPGA-ihoz és CPLD-ihez kifejlesztett ingyenes szoftver, amely az Internetről, a cég weboldaláról szabadon letölthető. A fejlesztőrendszer tartalmazza mindazon elemeket, amelyek kapcsolási rajz alapú, állapotgép alapú, illetve hardverleíró nyelv alapú fejlesztéshez szükségesek. Szimulációra korlátozottan alkalmas a fejlesztőrendszer, de ingyen letölthető a ModelSim XE, amely a Xilinx eszközökre szűkített képességű nagy tudású szimulációs eszköz.

A segédlet megírásakor a Xilinx ISE WebPack legfrissebb elérhető verziója a 9.1-es. (A fejlesztőrendszer állandó bővítés és fejlesztés alatt van.) A mérési segédletben ennek tulajdonságait vesszük alapul.

A mérési segédlet **bevezetést** kíván nyújtani – az FPGA-k tervezésébe, célja **a tervezési lehetőségek minél szélesebb körben** való ismertetése. Elmélyülésre – a korlátozott laboridő miatt – saját munkával, illetve a konzultációs időpontokban van lehetőség. A mérés sikeres elvégzését nagymértékben segíti a mérést előkészítő előadás anyagának elsajátítása.

A mérés során a SPARTAN-3 típust tartalmazó **demonstrációs panelt** (SPARTAN-3 Starter Board) fogjuk használni. A panelen alkalmazott SPARTAN-3 FPGA egy 256 lábú BGA tokban kapott helyet, kapuszáma 200000, de a kereskedelemben többféle kapuszámmal és tokozásban kapható. A panelen helyet kapott számos kiegészítő egység: 8 db. LED, egy 4 digites 7 szegmenses kijelző, 8 db. kapcsoló, 4 db. nyomógomb, PS2, RS232, és VGA portok, 2db. 256kx16bit SRAM, konfigurációs FLASH, és egy 50MHz-es órajel-forrás. Az FPGA összes I/O lába a panel peremén elhelyezett csatlakozókra van kivezetve. A konfigurációs bitminta **letöltésére** a laboratóriumban egy **JTAG letöltő kábel** szolgál.

A fejlesztőkörnyezet elindítása a *Start* menüből (*Xilinx ISE 9.1*), vagy közvetlen link segítségével történhet. A program elindítása után a következő felülettel találkozunk:

🔤 Xilinx - ISE	
File Edit View Project Source Process	
8 🗅 🖻 🖥 🕹 8 X 🖻 Ĝ 🗙 📨 🕬 🖸 8 🗩 🔎 🕤 🔊	🔁 E 🛛 🗔 🌽 😽
# M 🕷 🔄 💽 🖬 🖓 # 🗊 🖉 🖉 🗐 🏟 🏗 🔪 🚽 📜 👘	
(3)	
	http://www.xilinx.com
	×
	2
Console 🖸 Errors 🔝 Warnings Tot Shell 🕅 Find in Files	
	CAPS NUM SCRL

1. ábra A Xilinx ISE Webpack kezelőfelülete

A felhasználói felület felosztása:

- 1. Menüsor
- 2. Fontosabb parancsikonok
- 3. Munkafelület
- 4. Eszköztárak (ki/be kapcsolhatóak, helyük változtatható)

2. Mintafeladat megoldása – kapcsolási rajz alapú tervezés

A Xilinx ISE WebPack 9.1 segítségével hozzunk létre egy egyszerű példaprogramot a gyakorlópanelen található négydigites hétszegmenses kijelző és a LED-ek tesztelésére! A feladat első részét kapcsolási rajz alapon, a második részét állapotdiagramos tervezéssel mutatjuk be.

Első lépésként készítsünk el egy **új projekt fájlt** (*File→New Project*). Amennyiben rendelkezünk már projekt fájllal nyissuk meg az *Open project* menüponttal! A fejlesztőkörnyezet **minden projektnek külön könyvtárat** hoz létre, és a "Xilinx91i" mappába menti. Projektünk neve legyen "*vezetéknevünk_proba*" (javaslat), **legmagasabb** szintű forrásként kapcsolási rajz alapú (schematic) típust adjunk meg!

🔤 New Project Wizard - Create New Project		
Enter a Name and Location for the Project		
Project Name:	Project Location	
Veznev_proba	H:\FPGA\Xilinx91i\Veznev_proba	
Select the Tupe of Topul evel Source for the Project		
Top-Level Source Type:		
Schematic		~
More Info	< Back Next >	Cancel

2. ábra Új projekt létrehozása

Az általunk elkészíteni kívánt program modulokból épül fel, minden modulnak háromféle forrása lehet:

- Kapcsolási rajz (Schematic)
- Hardverleíró nyelv (VHDL, Verilog)
- Állapotdiagram (StateCad)

A legfelső, úgynevezett TOP forrásban fogjuk összeállítani modulunkat – ezt a legegyszerűbben kapcsolási rajz (schematic) típus esetén tudjuk megtenni!

A következő ablakban annak a félvezető eszköznek a tulajdonságait állíthatjuk be, amelyre fejlesztünk:

155	🛛 New Project Wizard - Device Properties							
ſ	Select the Device and Design Flow for the Project							
	Property Name	Value						
	Product Category	All						
	Family	Spartan3						
	Device	×C3S200 💌						
	Package	FT256						
	Speed	-5						
	Top-Level Source Type	Schematic 😪						
	Synthesis Tool	XST (VHDL/Verilog)						
	Simulator	ISE Simulator (VHDL/Verilog)						
	Preferred Language	VHDL						
	Enable Enhanced Design Summary							
	Enable Message Filtering							
	Display Incremental Messages							
	More Info	< Back Next > Cancel						

ábra
 FPGA tulajdonságainak beállítása

Forrást a Next gombra kattintva hozhatunk létre. Ezt megtehetjük most, vagy akár a későbbiek folyamán a Project→New Source... menüponttal! (Egyenlőre ne hozzunk létre forrást!)

A következő ablakban **meglévő forrást adhatunk hozzá** a projekt fájlunkhoz. E lépés végrehajtására is van lehetőségünk a későbbiek folyamán a *Project* \rightarrow *Add Source*... menüponttal.

155	lew P	roject Wizard - Create New S	Source			
6	ireate a	a New Source				
	1	Source File	Туре		Remove	
Cr	Creating a new source to add to the project is optional. Only one new source can be created with the New Project Wizard. Additional sources can be created and added to the project by using the "Project->New Source" command.					
	Existing sources can be added on the next page.					
	More Ir	ifo		< Back Next	> Cancel	

4. ábra

Új forrás létrehozása

Miután ismét a *Next* gombra kattintottunk, a projektünk összefoglalója jelenik meg:

🔤 New Project Wizard - Project Summary
Project Navigator will create a new project with the following specifications:
Project: Project Name: Veznev_proba Project Path: H:\FPGA\Xilinx91i\Veznev_proba Top Level Source Type: Schematic
Device: Device Family: Spartan3 Device: xc3s200 Package: ft256 Speed: -5 Synthesis Tool: XST (VHDL/Verilog) Simulator: ISE Simulator (VHDL/Verilog) Preferred Language: VHDL Enhanced Design Summary: enabled Message Filtering: disabled Display Incremental Messages: disabled
Karal Kar Karal Karal Kar Karal Karal Kar Karal Karal Kar Karal Karal Kara



Projekt tulajdonságainak összegzése

Ellenőrizzük, a projekt, valamint az eszköz **beállításait**! Amennyiben mindent rendben találunk, kattintsunk a *Finish* gombra!

Hozzunk létre a projectünkhöz forrás fájlt (*Project* \rightarrow *New Source*...)! Legyen a forrásunk típusa schematic, a neve tetszőleges¹! Amennyiben nem történt meg automatikusan, adjuk hozzá a forrás fájlt a projektünkhöz (*Project* \rightarrow *Add Source*...)

Project File: Verrey_proba ise Current State: New Module Name: poba -	y Pro Moo Sages sages sages sages sages sages sat port te Report eport	oject File: odule Name: arget Device: oduct Version: opartition information was for eport Name rithesis Report anslation Report ap Report see and Route Report gen Report gen Report	Vezne proba x:3:22 iSE 9 iSE 9	w_probaise 00-58256 1.031 Generate	VEZNEV_PROBA VEZNEV_PROBA P Detailed F	Project Status Current State: • Errors: • Warnings: • Updated: Partition Summary		New Sze máj 16 16:37:59 2007
Project File: Verrev_probaire Current State: New Module Name: proba - Frors:	ints Pro Mo. Tar Pro sages sages te Messages es sages fra sages fra port te Report te Report eport	oject File: odule Name: srget Device: oduct Version: partition information was for partition information was for partition information was for eport Name enthesis Report anslation Report ap Report see and Route Rep atic Timing Report gen Report	Vezne proba xc3i2 ISE 9. und Status W Source Wizard - Selet	ev_proba.ise 00-5/t256 1.03i Generate	VEZNEV_PROBA P Detailed F	Current State: • Errors: • Warnings: • Updated: Partition Summary Partones		New Sze máj 16 16 37 59 2007
Module Name: poba • Eurors: Target Device: w:3:200.51256 • Warnings: Product Version: ISE 91.03* • Updated: stages stages escapes Detailed Reports Report Name Status Synthesis Report Image: Transition Report New Source Wizard - Select Source Type Plea and Route Rep BMM File Static Timing Report Image: Plea and Route Reports Image: Static Timing Report Image: State Diagram Image: Index Report Image: State Diagram Image: Index Report Image: State Diagram Image: V	ints Tar Pro sages sages es sages es sages Trar sages Trar port Stat te Report teport	adule Name: arget Device: oduct Version: partition information was for export Name export Name export Name export Name anslation Report ace and Route Rep ace and Route Report gen Report gen Report	proba kx3s2 liSE 3 und. Status wy Source Wizard - Selet	00-5#:256 1.03i Generate	VEZNEV_PROBA P Detailed F	Errors: Warnings: Updated: Vartition Summary		Sze máj 16 16:37:59 2007
Image: Second	ints Tar Pro segges segges les es es sagges rt res sagges rt port stat port stat leport leport	srget Device: oduct Version: > partition information was for eport Name mithesis Report anslation Report ap Report actor Timing Report gen Report	xc3a2 ISE 9. Status Status ew Source Wizard - Selec	00-5/t256 1.03i Generate	VEZNEV_PROBA P Detailed F	Warnings: Updated: Partition Summary Reports		Sze máj 16 16:37:59 2007
Product Version: ISE 91.03i • Updated: Sze máj 16 16:37:59 2007 VEZNEV_PROBA Partition Summary No patition information was found. Detailed Reports Detailed Reports Beport Name Status Synthesis Report Translation Report New Source Wizard - Select Source Type Infos Status Beport Status Diagram Status Diagram </td <td>Pro sages sages te Messages es sages at port te Report leport</td> <td>oduct Version: partition information was for eport Name rithesis Report anslation Report so e and Route Rep atic Timing Report gen Report</td> <td>und Status</td> <td>1.03i Generate</td> <td>VEZNEV_PROBA P Detailed F</td> <td>Updated: Partition Summary Reports</td> <td></td> <td>Sze máj. 16 16:37:59 2007</td>	Pro sages sages te Messages es sages at port te Report leport	oduct Version: partition information was for eport Name rithesis Report anslation Report so e and Route Rep atic Timing Report gen Report	und Status	1.03i Generate	VEZNEV_PROBA P Detailed F	Updated: Partition Summary Reports		Sze máj. 16 16:37:59 2007
seages de Messages ges ges ges ges ges ges ges	sages sages te Messages es sages at Plac port te Report leport	eport Name rithesis Report anslation Report so e and Route Rep atic Timing Report gen Report	und. Status ew Source Wizard - Selec	Generate	VEZNEV_PROBA P Detailed F	Partition Summary Reports		
stages stages per per per per per per per per	eages sages at term may be sages at the Report state Report state sages sages states sages sages states sages sage	eport Name rithesis Report anslation Report ap Report ace and Route Rep atic Timing Report gen Report	und Status Status ew Source Wizard - Selec	Generate	Detailed F	Reports		
Detailed Reports Report Name Status Generated Errors Warnings Info Synthesis Report Image: Anchitecture Wizard - Select Source Type Map Report Image: Anchitecture Wizard - Select Source Type Image: Anchitecture Wizard - Select Source Type Image: Anchitecture Wizard - Select Source Type Map Report Image: State Timing Report Image: Anchitecture Wizard - Select Source Type Image: Anchitecture Wizard - Select Source Type Map Report Image: State Timing Report Image: State Timing Report Image: State Type State Timing Report Image: State Type Image: State Type Image: Type State Type Image: State Source Type Image: Type Image: Type State Type Image: Type Image: Type Image: Type Isource Image: Type Image: Type Imad	sages Beg Syn Trar Mag Piac Stat Bitg	epott Name whesis Report anslation Report ap Report ace and Route Rep atic Timing Report gen Report	Status ew Source Wizard - Selec	Generate	Detailed F	Reports		
Beport Name Status Generated Warnings Infos Synthesis Report Investigend Infos Infos Transition Report Investigend Infos Piece and Route Report Infos Infos Static Timing Report Infos Infos Bigen Report Infos Infos Verify Report Infos Infos Verify Report Infos Infos Verify Report Infos Infos State Diagram Infos Infos Verify Report Infos Infos Verify Repor	ges Reg Syn Tra Map Plac Stat Bitg	eport Name nthesis Report anslation Report ap Report sce and Route Rep atic Timing Report gen Report	Status ew Source Wizard - Selec	Generate	d I	-		
se source Wizard - Select Source Type	ss Syn Mag Plac Stat t	nthesis Report	w Source Wizard - Selec			Errors	Warnings	Infos
Translation Report Image New Source Wizard - Select Source Type Map Report BMM File Static Timing Report Image New Source Wizard - Select Source Type Bigen Report Image New Source Wizard - Select Source Type Image Neport Image Neport Image Neport Image Ne	Trar Map Plac Stat Bitg	anslation Report ap Report acce and Route Rep atic Timing Report	w Source Wizard - Selec					
Map Report Place and Route Rep Static Timing Report Bigen Report Bigen Report Static Timing Report Bigen Report Static Subject Place Solution Constraints File Static Solution Constraints File Static Solution Constraints File State Document Used Document	ort Map	ap Report ace and Route Rep atic Timing Report	PMM File	t Source Type				
Place and Route Rep Static Timing Report Bigen Report point In plementation Constraints File State Diagram In plementation Constraints File In plementation Constraints File State Diagram In plementation Constraints File In plemen	port Plac	ace and Route Rep atic Timing Report	PMM File	er source Type				
State Timing Report Bigen Report File name: File nam	ort Bitg	atic Timing Report	Dimini Tile					
t Bigen Report Godernatic File File ame: State Dagam File name: State Dagam File name: User Document WHDL Module Location: WHDL Locay VHDL Locator: WHDL Locator: WH	t Bilg	igen Report	IP (Coregen & Architecture Wiza MEM File	ardj				
Implementation Constraints File File name: Test Banch WaveForm Implementation Constraints File User Document Implementation Constraints File Verilog Module Location: Verilog Module Location: Verilog Test Fikure H:YPFGAVdimv91NVeznev_proba VHDL Lobrage VHDL Lobrage VHDL Test Bench VAble More Info < Back		the second se	Schematic					
Summay Summay More Info Mo			Implementation Constraints File					
Summary ages ta		8	State Diagram Test Rench WaveForm		File name:			
Location: Whole Info Module Location: WHOL Module H-VFPGAV&InvS1NVeznev_probe VHDL Datage VHDL Test Bench More Info Add to project More Info Cancel			User Document					
Add to project More Info Key Cancel More Info Key Cancel Key Cancel Key Cancel Key Cancel Key Cancel Key Cancel			Verilog Module		Location:			
Summary ages as Add to project More Info			Venilog Test Fixture		H:\FPGA\Xiinx91i\Vez	mey proba		
ummary ges s More Info More Info S S S S S S S S S S S S S S S S S S S			VHDL Librarv					
Add to project More Info < Back	ary	- E - E - E - E - E - E - E - E - E - E	VHDL Package					
des \$ Add to project More Info Cancel			VHDL Test Bench					
Add to project More Info < Back. Next > Cancel	jes							
More Info < Back Next > Cancel	\$				Add to project			
More Info Cancel Cancel								
		Ma	ore Info		< Back	Next> Cancel		
	ching Design	n Summary".						
sching Design Summary".	_							
ing Design Summary".		1						
ing Design Summery".								
hing Design Summary".	🔥 Warnings	Tcl Shell 🛛 🕅 Fi	ind in Files					

6. ábra

Új forrás létrehozása

A *View* menüpontban **állítsuk aktívra** a források (*Sources*), és a folyamatok (*Processes*) ablakát! Ennek hatására ezek megjelennek baloldalon.²

¹ Mivel később ez lesz a legfelső szintű modul, célszerű a nevében szerepeltetni a "top" megjelölést, pl. "top_pelda" (szerkesztői megjegyzés)

² Az előzetes megjelenítési beállításoktól függően, nem biztos, hogy szükség van az ablakok aktiválására, illetve nem biztos, hogy a fent említett helyen jelennek meg. (szerkesztői megjegyzés)

Váltsunk kattintással a forrásunk (itt: proba.sch) fülére:



Váltás a forrásra

Kezdjünk hozzá a programozáshoz! Első programunk a következő funkciókat fogja megvalósítani:

- Bekapcsolja az egymás után a hétszegmenses kijelzők egyes szegmenseit
- Bekapcsolja egymás után a gyakorlópanelen lévő LED-eket
- Mindezt egy kapcsolóval leállíthatjuk menet közben, ilyenkor őrzi az előző állapotokat

Ehhez szükségünk van egy **órajel leosztó áramkörre**, amely az egyes szegmensek ill. LEDek bekapcsolásának időközét állítja be (a láthatóság érdekében). Amennyiben csak szimulációt kívánunk végezni, eltekinthetünk az órajel-osztó áramkör megépítésétől.

Az órajel leosztó áramkört kapcsolási rajz alapú szerkesztővel definiáljuk:

- Hozzunk létre egy új forrást: Project→New Source... (típusa: schematic, neve (példánkban: "idozito.sch"))³!
- Amennyiben nem történt meg automatikusan, adjuk hozzá ezt a forrást a projektünkhöz Project→Add Source! A további lépések során ügyeljünk arra, hogy a most létrehozott forrás füle legyen aktív!
- Válasszuk ki a szükséges alkatrészeket a baloldalon található Sources ablak Symbols fülének segítségével.

Az alkatrészek megkeresése történhet kategóriák alapján, de használhatjuk a

³ Javasoljuk, hogy a könnyű tájékozódás miatt az elnevezésben szerepeljen a "oszto" v. "osztó" szó. Ügyeljünk, hogy ne a mostani, hanem az előzőekben létrehozott modul maradjon a TOP modul. Beállítás: jobb kattintás a forrás felett, majd *Set As Top Module (szerkesztői megjegyzés)*

névkeresőt is (*Symbol Name Filter*). Az alkatrészek forgatását az Orientation mezőben állíthatjuk be. (A panel órajele 50MHz - két 16 bites osztóval megvalósíthatjuk mind a pergésmentesítéshez, mind az időzítéshez szükséges belső órajelet. (A teljes osztási lehetőség kihasználásával a legalsó bit

 $f = \frac{50MHz}{2^{32}} = 11,6mHz$ frekvenciával billeg.)

Sources		×
Categories		
<all symbols=""></all>		
Arithmetic		
Buffer		
Carry_Logic		
Comparator		~
Counter		
Symbols		
acc16		
acc4		
acco add16		
add10 add4		
10		<u>~</u>
Symbol Name Filter		
Urientation		
Hotate U		×
	Symbol Info	
📭 Sources 👩 Sn	apshots 👔 Libraries	🔛 Symbols

8. ábra Az alkatrészek kiválasztása

4. Kezdjük el szerkeszteni a következő hálózatot (alkatrész elhelyezés, huzalozás)!



9. ábra

Az alkatrészek elhelyezése a szerkesztő felületen

Alkatrészek:

- CB16CE: Engedélyezhető 16 bites számláló (2db, frekvenciaosztáshoz)
- BUF: Nem invertáló puffer
- VCC: Pozitív tápfeszültség (magas szint (H))
- GND: Földpont (alacsony szint (L))

A hozzávezetéseket, illetve az **alkatrész összeköttetéseket** az *Add Wire* ikon () segítségével készíthetjük el. Jól látható, hogy a számláló kimenetéhez csatlakoztatott vezeték vastagabb a többinél, ez jelzi annak **busz** jellegét.

5. Végezzük el a hálózat teljes bekötését (be- és kimenetek (I/O Marker) hozzáadása, csatlakozás a buszra)!

Tegyünk az első számláló kimenetére egy, a második számláló kimenetére két buszleágazást (*Add Bus Tap*)⁴! Helyezzük el az I/O Markereket!



10. ábra

Buszcsatlakozók és I/O markerek elhelyezése

A felhasznált parancsok illetve parancsikonok:

• Select (): Kiválasztás/mozgatás.

⁴ Az *Options* ablakban beállítható, hogy huzalozáskor a buszra való bekötés automatikusan helyezzen el egy buszcsatlakozót (Tap) is. (szerkesztői megjegyzés)

- Add Wire (]]: Vezeték hozzáadása.
- Add Net Name (b): Vezetékezés elnevezése.
- Rename Selected Bus (📲): A kijelölt busz átnevezése.
- Add Bus Tap (>): Buszos vonalra csatlakozás hozzáadása.
- Add I/O Marker (🚬): Be-, kimeneti pontok hozzáadása.
- Add Symbol (🚉 -): Szimbólum (alkatrész, könyvtári, vagy saját) hozzáadása.

(Bármely művelethez a bal alsó ablakban (*Process/Options*) találunk beállítási lehetőségeket, illetve műveleteket.)

- 6. Nevezzük el a különböző busz-elnevezéseket és az I/O markereket!
 - Vezeték, busz elnevezése: Add Net Name (a bal alsó ablakban az Options fülnél a Name sorba írjuk a nevet, majd kattintsunk az elnevezni kívánt vezetékre. Az esztétikus kivitelezésre törekedve ügyeljünk, hogy a vezeték elég hosszú legyen a felirat elhelyezéséhez!).⁵
 - I/O marker elnevezése: dupla kattintás az I/O markeren, a felugró ablakban *Name* mező átírása.



11. ábra Az I/O markerek, vezetékek, buszok elnevezése

A buszok, illetve leágazásaik elnevezése utal arra, hogy adott buszon belül melyik

⁵ Az Options ablakban beállítható automatikus sorszámozás, amely hasznos, ha több, egy buszhoz tartozó, egymás után következő címkét kell elhelyeznünk. (szerkesztői megjegyzés)

vonalra (bitre) csatlakozunk (Pl. a fenti ábrán CLOCK(15:0) egy 16 bites busz, amelynek legfelső helyértékű bitjére csatlakozunk (CLOCK(15)). A markerek elnevezése tetszőleges⁶ (a fenti ábrán: CLK: clock-órajel, CLK24: az órajel frekvenciájának 24-ed része, CLK18: az órajel frekvenciának 18-ad része). A CLOCK18 jelét a későbbiekben használhatja a kapcsolók, vagy nyomógombok pergésmentesítésére.

 A fejlesztőkörnyezet lehetőséget nyújt saját alkatrész definiálására az átláthatóság megkönnyítése érdekében – használjuk ki ezt a lehetőséget! A *Tools* menüben válasszuk ki a *Symbol Wizardot*!

🔤 Symbol Wizard - S	ource Page			
-Pin Name Source				
Specify Manually				
 Using Schematic 	PROBA			~
🔿 Using Symbol				~
Import Symbol	Attributes			
Shape				
Do not Use Befere	nce Symbol			
Bectangle				
O Square				
Use Reference Syl	nbol			
			(Browse
· · · · · · · · · · · · · · · · · · ·				
		< Back	Next >	Cancel

12. ábra Saját alkatrész definiálása

Fontos, hogy a *Using Schematic*, illetve a *Rectangle* opció legyen bejelölve. A felhasznált kapcsolási rajz (vagy egyéb modul) nevének helyes beállítására ügyeljünk! (Azt a modult kell beállítani, amelyikből szimbólumot szeretnénk készíteni!)

⁶ A markerek elnevezésével célszerű utalni a jelek funkciójára. (szerkesztői megjegyzés)

A *Next*-re kattintva jelentkező következő ablakban **az alkatrész be és kimenetei** vannak megjelenítve, ezen nem szükséges változtatnunk. Megadhatjuk azt, hogy a szimbólum jobb, vagy baloldalán legyen-e az adott kivezetés (*Side*), illetve a megjelenítés sorrendjét is beállíthatjuk (*Order*).

Ismét továbblépve (*Next*) az alkatrészünk **geometriai tulajdonságait** módosíthatjuk (átméretezhetjük). Az utolsó ablakban megtekinthetjük alkatrészünk **előnézetét**.

🔤 Symt	ool Wizard	- Pin Page			🖾 Symbol Wizard - (Option Page	
Symbol I IDOZITI Pin Name CLK CLK24 CLK18	Vame Polarity Input Output Output	Side Left Right Right	Order 1 1 2	Add Pin Remove Pin/Spacer Insert Spacer Move Spacer Up Move Spacer Down	Symbol Name Font Size Pin Name Font Size Pin Length Pin Space Pin Edge Symbol Width Symbol Origin	56 1 24 1 64 1 32 1 256 1 Left Bottom 1	symbol name font size FD ui ui ui ui ui ui ui ui ui ui
Keyboar -Use arro -Use Sp. -Use Tal	d Usage Tips: www.eys to go ace to active o b to change fo	to previous/next cell in dotring a cell inside grid cus among controls	side grid	xt > Cancel		- K Back	Symbol origin Image: Next > Cancel

13. ábra

14. ábra

Saját alkatrész kivezetéseinek beállításai

Saját alkatrész méreteinek beállítása

8. Váltsunk a legfelső szintű modulunk fülére (itt: proba.sch)!

Amennyiben mindent jól csináltunk, saját definiálású alkatrészünknek látszania kell a szimbólumok között:

Sources	×
Categories <-All Symbols> <h: end<br="" fpga="" xilinx91i="">Arithmetic Buffer Carry_Logic</h:>	DEMO>
Comparator Symbols	<u> </u>
1002110	

15. ábra

Saját alkatrész kiválasztása

9. A legfelső szintű modulban készítsük el a következő áramkört!⁷





A végleges áramkör a szegmensek tesztelésére⁸

⁷ Ügyeljünk arra, hogy azt a modult, amelyben a saját alkatrészt definiáltuk, ne írjuk felül, vagy ne töröljük! Ebben az esetben – bár a rajz alapján létrehozott alkatrész lehelyezhető, de – a fordítás során hibajelzést kapunk, hiszen nem tudja a fordító "visszafejteni" a szimbólumot. (szerkesztői megjegyzés)

⁸ A CLOCK és SW0 bemenetekre IBUF, a DISP(7:0) kimenetekre pedig OBUF (8 bites OBUF8) elemeket kell elhelyezni, ez a szabályos megoldás. Ettől függetlenül működhet így is az elrendezés (az automatikus bufferelhelyezés miatt), de a be- és kimenetek nincsenek optimálisan bufferelve. Az IBUF és OBUF elemekkel számos beállítás mellett pl. a logikai családhoz, illetve a logikai szintekhez való illesztés valósítható meg. Számos célra állnak rendelkezésre buffer elemek, ezekről többek között a súgóból tájékozódhatunk. A RESET címke egy megfelelő bufferen keresztül a RESET bemenethez csatlakoztatható. (szerkesztői megjegyzés)

A felhasznált elemek:

- I/O markerek
 - CLOCK (órajel)
 - DISP(7:0) (7 szegmens)
 - SW0 (kapcsoló)
- IDOZITO (saját alkatrész)
- BUF; OBUF8 (bufferek)
- CB4CE (számláló)
- D3_8E (dekóder)
- OR2 (kétbemenetű VAGY kapu)
- VCC (pozitív tápfeszültség, H szint)
- Adjunk a projektünkhöz egy Implementation Constraints fájlt (*Project→New* Source), amelyben a lábkiosztást fogjuk definiálni!⁹

Kétféle módon tudjuk a lábakat az egyes I/O markerekhez rendelni: karakteres, illetve grafikus módon.

Lábkiosztás grafikusan:

- Tegyük aktívvá az Imlementation Constraits fájlunkat!
- Kattintsunk duplán a Create Area Constraints pontra!¹⁰

A grafikus felület megjelenése előtt a fordító elvégzi a felépített áramkör **szintaktikai ellenőrzését**, ezek után a következő ablak nyílik meg:

⁹ Ügyeljünk, hogy a fájl létrehozásakor a legfelső szintű modulhoz rendeljük azt. (szerkesztői megjegyzés)

¹⁰ A Create Area Constraints pont helyett az Assign Package Pins menüpont választása helyesebb, mert míg az előbbi a belső elrendezés kialakítását is befolyásolja, utóbbi a lábkiosztás meghatározására való. (A két felhasználói felület azonos.) (szerkesztői megjegyzés)

Xilinx PACE - H:\FPGA\Xilinx91i\ENDEMO\PINS.ucf		
File Edit View IOBs Areas Tools Window Help		
🛛 🗅 🖨 🖨 🗠 🗰 🦊 🧏 🛄 🔂 🔛 🕱	☞ 🎥 🔏 🗖 🖉 🖉 📟 🗆 💥 🔍 🍳 💥 🔍 📴 🔲 🗖 🗖 🗖 🗖	
🖹 Design Browser	Package Pins for xc3s200-5-ft256	
	Top View 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	1
		A
DISP(2)		В
DISP<0>		С
		D
🖆 Design Object List - I/O Pins 📃 🗖 🗙		E
I/O Name I/O Direction Loc Bank I/O Std.		F
DISP<0> Output		G
		н
DISP<3> Output		J
DISP<4> Output DISP<5> Output		К
DISP<6> Output		L
SVV0 Input		м
		N
# Group I/O Direction Loc I/O Std.		P
		R
		т
	h Berley Ban & Baltintur Man &	_ _

17. ábra Lábkiosztás grafikus felületen

A különböző **jelölések értelmezéséhez** bekapcsolhatjuk a *View* \rightarrow *Toolbars* \rightarrow *Legend* menüpontot. Ezt a funkciót a *CTRL*+*L* billentyűkombinációval is aktiválhatjuk.

A Design Browser ablakból **"húzd és vidd" módszerrel** vigyük át a jobb oldalon található ablak megfelelő helyére minden I/O markert! Miután az összes markert elhelyeztük, **mentsük el** a fájlt, majd **zárjuk be**!

A lábkiosztásra lehetőséget nyújt egy **szövegszerkesztő** is, melyet az *Edit Constraints*-re kettőt kattintva hívhatunk elő.



Lábkiosztás szöveges módszerrel

```
#PACE: Start of Constraints generated by PACE
 1
 2
 3 #PACE: Start of PACE I/O Pin Assignments
 4 NET "CLOCK" LOC = "T9" ;
 5 NET "DISP<0>" LOC = "E14"
6 NET "DISP<1>" LOC = "G13"
                                           ;
                                           ;
 7 NET "DISP<2>" LOC = "N15"
8 NET "DISP<3>" LOC = "P15"
9 NET "DISP<4>" LOC = "R16"
10 NET "DISP<5>" LOC = "F13"

        11
        NET "DISP<6>"
        LOC
        = "N16"

        12
        NET "DISP<7>"
        LOC
        = "P16"

                                           ;
                                           ;
13 NET "SWO" LOC = "F12" ;
14
     #PACE: Start of PACE Area Constraints
15
16
     #PACE: Start of PACE Prohibit Constraints
17
18
19
     #PACE: End of Constraints generated by PACE
20
```



A lábkiosztás szöveges formája

A #-tel kezdődő sorokat a fordító **nem veszi figyelembe**, így lehetőségünk van univerzális lábkiosztást írni, melyben csak az adott programban használt perifériákat használjuk, a többit #-tel megjelöljük.

11. Hozzuk létre a konfigurációs bitmintát! Kattintsunk jobb egérgombbal a *Generate Programming File*-ra, majd a *Properties*-re:

Process Properties		X		
Category				
General Options Configuration Options	Readback Options			
Readback Options				
	Property Name	Value		
	Security	Enable Readback and Reconfiguration 🛛 💌		
	Create ReadBack Data Files			
	Allow SelectMAP Pins to Persist			
	Create Logic Allocation File			
	Create Mask File			
	Property	display level: Standard 💌 Default		
	ОК	Cancel Apply Help		



A programozó fájl elkészítésének beállításai

A *ReadBack Optionsban* jelöljük be a *Create Readback Data Files*, valamint a *Create Mask File* opciót, majd nyomjuk meg az OK gombot! (Amennyiben ezt a lépést kihagyjuk, nem fog működni a *Verify* funkció)!

Kattintsunk duplán a Generate Programming File-ra, majd azon belül a Configure Device (iMPACT)-re!



21. ábra Az iMPACT elindítása

A Configure devices using Boundary-Scan (JTAG) (Eszközök konfigurálása

peremfigyelés segítségével) legyen kijelölve, majd kattintsunk a befejezésre (*Finish*)!

🔤 iMPACT - Welcome to iMPACT				
Please select an action from the list below				
 Configure devices using Boundary-Scan (JTAG) 				
Automatically connect to a cable and identify Boundary-Scan chain 💌				
O Prepare a PROM File				
O Prepare a System ACE File				
O Prepare a Boundary-Scan File				
SVF 💟				
O Configure devices				
using Slave Serial mode				
< Back Finish Cancel				



Csatlakoztatás a peremfigyelés módszerével

Ezután **automatikusan** megnyílik a **következő ablak.** (Ha mégsem, akkor jobb egérgombbal kattintsunk az IC-n, majd válasszuk az *Assign New Configuration File*-t!) **Válasszuk ki** a betöltendő fájlunkat (*.bit* kiterjesztéssel), majd nyissuk meg (*Open*)!

A következő ablakban egy, az FPGA-hoz tartozó külső konfiguráló **FLASH memóriához** (IC9, XCF02S) kell **rendelni az azt leíró gyári fájl**t (xcf02s.bsd).¹¹

¹¹ Bár példánkban nem a külső FLASH memóriába történik a konfigurációs bitminta betöltése, de a letöltés ellenőrzéséhez szükség van erre a lépésre, és a következő XCF02S-sel kapcsolatos lépésekre is. Másik (javasolt) megoldás a betöltési ablakban (ld. 23. ábra) a Bypass választása, ilyenkor a nem használt JTAG elem lényegében nem kerül a peremfigyeléses láncba. Így nem kell hozzá fájlt kiválasztani, törölni sem. (szerkesztői megjegyzés)



23. ábra

Az XCF02S memória gyári leíró fájljának kiválasztása

A jobb oldali "IC"-n jobb egérgombbal kattintva töröljük az XCF02S FLASH memóriát (*Erase*)!



24. ábra

A konfigurációs FLASH memória törlése

A bal oldali "IC"-n kattintva **töltsük be** a létrehozott konfigurációs bitmintánkat az FPGA-ba (*Program*)! (Ha változtatunk valamelyik modulon, a bitminta generálást mindig meg kell ismételni!)



25. ábra Program letöltése az FPGA-ba

3. Feladatok a kapcsolási rajz alapú tervezéshez

- 1. Figyelje meg a működést, javítsa a hibát!
- 2. Alakítsa át az elrendezést úgy, hogy a BTN3 nyomógomb hatására a kapcsolás kerüljön alaphelyzetbe (a BTN3 nyomógombot pergésmentesítse)!
- 3. Alaphelyzetben minden kijelző be van kapcsolva. Bővítse a kapcsolást, hogy egyszerre mindig csak egy kijelző legyen bekapcsolva. A kiválasztást a BTN2 nyomógombbal lehessen megtenni!

Segítség a feladat megoldásához:



26. ábra

Az első feladat egy lehetséges megoldása¹²

¹² A kimeneteken kimeneti bufferek (OBUF), a bemeneteken pedig bemeneti bufferek (IBUF) elhelyezése szükséges! (szerkesztői megjegyzés)

4. Mintafeladat kiegészítése állapotdiagram modellel

A 2. fejezetben definiált feladatból a LED-ek egymás után történő bekapcsolását **állapotdiagramos tervezéssel**, a StateCad program segítségével oldjuk meg. (A részfeladat külön funkcióként való megvalósításához nyithat új projektet, de itt – a feladatkiírás szerint – az előző tervben implementáljuk a megoldást.) Az állapotdiagramos tervezésnél **állapotokat definiálunk**, valamint megadjuk az egyes állapotok közötti **átmenet feltételét**.

- Hozzon létre egy State Diagram típusú új forrást (Project→New Source)! A fejlesztőkörnyezet automatikusan megnyitja az állapotdiagram-szerkesztőt.
- 2. A Draw State Machines ikonra kattintva hozza létre az alábbi geometriai elrendezést!



27. ábra

Az állapotgép geometriai elrendezése

A *Next*-re kattintva az alaphelyzetbe állítás szinkronitását állíthatjuk be. Válasszuk az **aszinkron** módot!





A Reset feltétel szinkronitásának megadása

A Setup Transitions ablakban jelöljük be a Next-et és a Previous-t, és válasszuk az !UP, illetve UP funkciót!

Design Wizard : Setup Transitions 🛛 🛛 🔀				
Each state can h	Sample			
next state and pre shows the effects	eri as transitions going from it to the evious state. The sample window s of your selections			
To place the stat	e machine, click Finish. Move the			
cursor to the desi button.	ired location and click the left mouse			
Add Transitions	Sot condition to	NEXT PREV		
Loop back:	@ELSE			
💌 Next:	IUP			
Previous:	UP	Default		
Help	K Back Finish	Lancel		



Az állapotgép futásának beállítása

3. Az egyes állapotokon kettőt kattintva hozzuk elő az Output Wizard-ot!



30. ábra

Az egyes állapotok beállítása

 $|\times|$

Az **állapotokat konstansként** adjuk meg. Az **elnevezés tetszőleges**, de természetesen nem szerepelhet két különböző állapotnál ugyanaz a név. Az **adatszélesség 8 bit**, mivel 8 kimenetet szeretnénk vezérelni. **Írjuk be** binárisan az általunk kívánt értéket minden állapothoz!

Edit State

Logic Wizard		X
And Bit Compare Buffer Constant	A constant value Constants may b (default or in a b. To use another t	e is assigned. e in decimal ase you specify). base, prefix the
Customize		
0 _		
		Data path width
Help	OK Cancel	Registered



State Name:	STATEO				
Outputs: Una	Outputs: Unassigned outputs are retained (i.e. q=q)				
Q = ^b000000	101;				~
					~
Output Wiza	ard Crea	te counters	, muxes, et	c. with the w	izard.
– Justify State	Name —		_Justify O	utput	
C Left 🖲	Center () Right	C Left	Center	C Right
		OK	Cancel		Help



- A kimenet típusának és szélességének megadása
- 4. Ezután kattintsunk az *Optimize* ikonra (itt több ablakon keresztül beállíthatjuk, hogy a fordító milyen szempontok szerint **optimalizálja** az állapotgépet)! Céleszközként *FPGA*-t, célforrásként *VHDL*-t válasszunk!
- 5. Kattintsunk a Generate HDL ikonra, majd sikeres lefordítás után adjuk hozzá a projectünkhöz a *.dia, és a *.vhd fájlokat a projekt Add Copy of Source menüpont segítségével! (Amennyiben nem vagyunk biztosak az állapotgép helyes működésében, szimulálhatjuk a StateBench segítségével.)
- 6. Ha minden lépést helyesen csináltunk végig, megjelenik a két fájl a forrásaink között.



33. ábra

StateCad fájl, és annak VHDL forrása a projektben

7. Hozzunk létre saját alkatrészt (saját kapcsolásrajzi szimbólumot) a *.vhd fájlunkhoz!



34. ábra

Szimbólum rendelése a VHDL forráshoz

8. Valósítsuk meg a következő áramkörrészletet! (PEREG és ORA az előzőekben létrehozott órajel osztó CLK18 és CLK24 kimenetei)



35. ábra A kiegészítő áramkör rajza¹³

¹³ A be- és kimeneteken IBUF és OBUF elemek elhelyezése szükséges. (szerkesztői megjegyzés)

5. Feladatok az állapotdiagram alapú tervezéshez

- 1. Próbálja ki SW1 kapcsoló hatását!
- 2. **Rajzolja át** a kapcsolást, illetve az állapotdiagramot úgy, hogy a fény oda-vissza fusson, és az irányt az SW2 kapcsolóval lehessen beállítani!
- Hozzon létre egy olyan logikát, amely oda-vissza futtat egy fénypontot a LED soron, de azok a LED-ek nem világítanak, amelyek hozzárendelt SW kapcsolója aktív! (pl.: SW3 és SW4 aktív, akkor LD3 és LD4 ne világítson.)
- 4. Az előző **két programot egyesítse** úgy, hogy a BTN2 nyomógombbal lehessen közöttük!
- 5. Valósítson meg Knight-Rider futófényt a LED-eken!*

BMF KVK MAI

6. VHDL modul fejlesztése

Az eddig megszerzett ismereteinket bővítsük **VHDL programozással**! Írjunk egy egyszerű VHDL programot, mely a bemenetére érkező 4 bites bináris számból előállítja annak 7-szegmenses kódját!

7 szegmenses kódtáblázat közös anódos kijelzőre:

Hexadecimális jegy	Bináris kód	7 szegmenses kód Dpgfedcba
0	0000	11000000
1	0001	11111001
2	0010	10100100
3	0011	10110000
4	0100	10011001
5	0101	10010010
6	0110	1000010
7	0111	1111000
8	1000	10000000
9	1001	10010000
А	1010	10001000
b	1011	1000011
С	1100	11000110
d	1101	10100001
E	1110	10000110
F	1111	10001110





1. Hozzunk létre egy új projektet, az elsődleges forrás legyen ismét schematic!





A projekt és a legfelső szintű forrás létrehozása

2. Új forrásként adjunk a projektünkhöz egy VHDL típusú fájlt!

🈋 State Diagram	File name:
A Test Bench WaveForm	KOD_SEGM
Verilog Module	Location:
Will VHDL Module	H:\FPGA\Xilinx91i\HET_SEGM
VHDL Library	
WHDL Fackage	

38. ábra



3. Adjuk meg a VHDL forrás be- és kimeneteit, valamint azok szélességét:

15	📧 New Source Wizard - Define Module					
	Entity Name Architecture Name	KOD_SEGM Behavioral				
	Port Name	Direction	Bus	MSB	LSB	~
	IN_BIN	in	✓ ✓		3	0
	OUT_SEGM	out	✓		6	0
		in	× 🗌			



A VHDL modul inicializálása

Felhasznált be- és kimenetek:

- Bemenet (IN_BIN): 4 bites bináris kód
- Kimenet (OUT_SEGM): 7 szegmenses kód (7 bites, mivel példánkban a tizedespontot nem használjuk)

A következő ablakban egy összegzés látható a beállításainkról, ez alapján a program létrehozza a VHDL modul inicializálását, nekünk már csak **a funkcionális programot kell megírnunk**.

```
Company:
  Engineer:
-- Create Date: 09:04:29 05/29/2007
- Design Name:
                KOD_SEGM - Behavioral
 - Module Name:
  Project Name:
 - Target Devices:
 - Tool versions:
 - Description:
- Dependencies:
-- Revision:
- Revision 0.01 - File Created
-- Additional Comments:
___
 _____
library IEEE;
use IEEE.STD LOGIC 1164.ALL;
use IEEE.STD LOGIC ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
---- Uncomment the following library declaration if instantiating
---- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;
entity KOD SEGM is
   Port ( IN_BIN : in STD_LOGIC_VECTOR (3 downto 0);
          OUT_SEGM : out STD_LOGIC_VECTOR (6 downto 0));
end KOD SEGM;
architecture Behavioral of KOD_SEGM is
begin
end Behavioral;
```

40. ábra

A fejlesztőkörnyezet által generált VHDL kódváz

4. Írjuk be a 7 szegmenses dekódolót megvalósító kódrészletet a programvázba! (A VHDL kódot a "*begin*" és az "*end Behavioral;*" közé írjuk be. Amennyiben "--" (két mínusz jel) kerül a kódba, a fordító nem veszi figyelembe a sor végéig álló szöveget (megjegyzések, jegyzetek lehetősége).¹⁴)

¹⁴ VHDL példákat és leírást az előadáson, illetve a kiadott segédletekben találhat. (szerkesztői megjegyzés)

begin				
with IN_BIN	SELect			
OUT_SEGM<=	"1111001"	when	"0001",	 1
_	"0100100"	when	"0010",	 2
	"0110000"	when	"0011",	 3
	"0011001"	when	"0100",	 4
	"0010010"	when	"0101",	 5
	"0000010"	when	"0110",	 6
	"1111000"	when	"0111",	 7
	"0000000"	when	<i>"</i> 1000",	 8
	"0010000"	when	<i>"</i> 1001 <i>"</i> ,	 9
	"0001000"	when	<i>"</i> 1010 <i>"</i> ,	 A
	"0000011"	when	<i>"</i> 1011 <i>"</i> ,	 b
	"1000110"	when	″1100″ ,	 С
	"0100001"	when	<i>"</i> 1101 <i>"</i> ,	 d
	"0000110"	when	<i>"</i> 1110 <i>"</i> ,	 E
	"0001110"	when	"1111",	 F
	"1000000"	when	others;	 0
<mark>end</mark> Behavioral;	:			

41. ábra Az általunk írt VHDL kód

A fenti kódsorozat egy **switch-case szerkezet** (jobb oldalt a gerjesztés, bal oldalt az erre adott válasz látható).

5. Készítsünk kapcsolási rajz szimbólumot VHDL forrásunkból!





Kapcsolási rajz szimbólum létrehozása VHDL forrásból

Ellenőrizzük, hogy alkatrészünk megjelent-e a szimbólumok között!

Sources	×
Categories	
<all symbols=""></all>	~
<h: fpga="" het_segm="" xilinx91i=""></h:>	
Arithmetic	
Buffer	
Carry_Logic	
Comparator	
Symbols	
KOD_SEGM	

43. ábra

Elkészített szimbólumunk a projekt szimbólumai között

6. **Hozzuk létre** a feladatnak megfelelő logikai elrendezést a TOP modulban! A program egy **lehetséges megvalósítása** az alábbi ábrán látható.



44. ábra A feladat egy lehetséges megoldása¹⁵

Az első két blokkot helyettesíthetjük az első feladat során megalkotott órajel osztó áramkörrel. Az eredeti feladatkiírás teljesítéséhez rögtön az első projektben is implementálhatjuk VHDL modulunkat a gyorsabb fejlesztés érdekében. (Így nincs szükség a kész kapcsolás átmásolására.)

¹⁵ A CLK bemeneten OBUF elem elhelyezése szükséges. (szerkesztői megjegyzés)

7. Feladatok a VHDL alapú tervezéshez

- Módosítsa a logikát, hogy egy kapcsolóval (SW0) lehessen kiválasztani, hogy felfelé, vagy lefelé számlálás történjen!
- 2. Módosítsa a logikát, hogy egy másik kapcsolóval (SW1) át lehessen állítani a számlálást decimálisra!
- 3. Készítsen 2 vagy több digitre számlálót!
- 4. Valósítsa meg saját ötletét, vagy a mérésvezető által kiadott feladatot! (Nyomógomb lenyomások számlálása, aritmetikai egység, stb.)

1. sz. melléklet – Az általunk használt gyakorlópanel lábkiosztása

Periféria	Pozíció	Megjegyzés
SW0	F12	Kapcsoló
SW1	G12	Kapcsoló
SW2	H14	Kapcsoló
SW3	H13	Kapcsoló
SW4	J14	Kapcsoló
SW5	J13	Kapcsoló
SW6	K14	Kapcsoló
SW7	K13	Kapcsoló
BTN0	M13	Nyomógomb
BTN1	M14	Nyomógomb
BTN2	L13	Nyomógomb
BTN3	L14	Reset nyomógomb
DISP_LED0	E14	a
DISP_LED1	G13	b
DISP_LED2	N15	c
DISP_LED3	P15	d
DISP_LED4	R16	e
DISP_LED5	F13	f
DISP_LED6	N16	g
DISP_LED7	P16	dp
AN0	D14	0. digit anódja
AN1	G14	1. digit anódja
AN2	F14	2. digit anódja
AN3	E13	3. digit anódja
LED0	K12	
LED1	P14	
LED2	L12	
LED3	N14	
LED4	P13	
LED5	N12	
LED6	P12	
LED7	P11	
CLOCK	Т9	Órajel

2. sz. melléklet – Néhány egyszerű VHDL példa¹⁶

1. NAND kapcsolat kétféle formában

```
library ieee;
use ieee.std_logic_1164.all;
   _____
entity NAND_gate is
port( A: in std_logic;
    B: in std_logic;
     Y: out std_logic);
end NAND_gate;
 -----
architecture NAND_leiras of NAND_gate is
begin
     process(A, B)
     begin
          if (A='1' \text{ and } B='1') then
          Y <= '0';
     else
          Y <= '1';
     end if;
     end process;
end NAND leiras;
```

¹⁶ E melléklet csak érintőleges betekintést ad a VHDL programozásba. Az Interneten számos példagyűjtemény és részletes leírás fellelhető. (szerkesztői megjegyzés)

2. 2-ről 4-re dekódoló kétféle formában

```
library ieee;
use ieee.std_logic_1164.all;
-----
entity DECODER is
end DECODER;
_____
architecture leiras of DECODER is
begin
    process (I)
    begin
    case I is
        when "00" => O <= "0001";
        when "01" => 0 <= "0010";
        when "10" => O <= "0100";
        when "11" => O <= "1000";
        when others => 0 <= "XXXX";
    end case;
    end process;
end leiras;
```

3. D flip-flop

```
library ieee;
use ieee.std_logic_1164.all;
-----
entity dff is
port( D: in std_logic;
    CK: in std_logic;
   Q: out std_logic);
end dff;
-----
architecture leiras of dff is
begin
    process(D, CK)
    begin
    if (CK='1' and CK'event) then
      Q <= D;
    end if;
    end process;
end leiras;
```

4. 4 bites shift-regiszter

```
library ieee;
use ieee.std_logic_1164.all;
_____
entity shift_reg is
port( I: in std_logic;
    CK: in std_logic;
    Q: out std_logic_vector(3 downto 0));
end shift_reg;
-----
architecture shift_leiras of shift_reg is
    signal S: std_logic_vector(3 downto 0):="1111";
begin
    process(I, CK, S)
    begin
    if CK'event and CK='1' then
         S <= I & S(3 downto 1);</pre>
    end if;
    end process;
    Q <= S;
end shift_leiras;
```

5. 8 bites számláló

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
_____
entity szamlalo is
port( CK: in std_logic;
     CL: in std_logic;
     Q: out std_logic_vector(7 downto 0));
end counter;
-----
architecture szlo_leiras of szamlalo is
     signal Pre_Q: std_logic_vector(7 downto 0);
begin
     process(CK, CL)
     begin
     if CL = '1' then
          Pre_Q <= Pre_Q - Pre_Q;</pre>
     elsif (CK='1' and CK'event) then
          Pre_Q <= Pre_Q + 1;</pre>
     end if;
     end process;
     Q <= Pre_Q;
end szlo_leiras;
```