University of Pannonia Dept. Of Electrical Engineering and Information Systems



Laboratory 2: Addings Solution

Appendix for using Xilinx EDK/SDK 10.1 SP3

Instructor: Zsolt Vörösházi, PhD.



East material exempt per Department of Commerce license exception TSU



SDK 10.1 – Step 1)

Run Xilinx Platform Studio SDK



Select workplace / XPS Project directory

ŝ	Xilinx Platform Studio SDK	×			
ſ	Select the XPS design to develop applications XPS Project: D:\FPGA\BEAGYAZOTT_RENDSZEREK\10_1\02_LAB\system. Vote)			
SDK provides an intuitive environment for configuration of embedded software platforms and development of embedded applications. Press F1 for more detailed information.					
	OK Cancel				

Select Application Wizard

Create a new SDK Application project

🏫 Application Wizard		
Select a Wizard		1
Create a new C application project, and a point unless you have already created yo	allow SDK to manage its Makefile. This is the typical starting our project.	
Wizards:		
Create a New SDK C Application Proje Import XPS Application Projects	ect	
Import an Existing SDK Application Pro	oject into this Workspace	
Do not launch Application Wizard at st	tartup	
	< Back Next > Finish Cance	el
-		

Add project name

 Add project name: "DipTest" and processor instance "MicroBlaze_0"

Project Name: DipTest Processor: microblaze_0 (MICROBLAZE) Image: Create Sample File in Project Image: Create Sample File in Project Image: Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse	🏫 New Pi	roject	X
Project Name: DipTest Processor: microblaze_0 (MICROBLAZE) ✓ Create Sample File in Project ✓ Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes			ø
Project Name: DipTest Processor: microblaze_0 (MICROBLAZE) Image: Create Sample File in Project Image: Create Sample File in Project Image: Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse			
Processor: microblaze_0 (MICROBLAZE) ✓ Create Sample File in Project ✓ Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse	Project Na	ame: DipTest	_
Create Sample File in Project Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse	Processor	r: microblaze_0 (MICROBLAZE)	
Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse,	🔽 Create	e Sample File in Project	
Use Default Location for Project (recommended) Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse,	_		
Location: D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse	Use D	Default Location for Project (recommended)	
	Location;	D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/DipTes Browse	

Add project type

- Project type: Xilinx MicroBlaze Executable
- Configurations: Debug / Release / Profile

🟫 New Project	
	S
Project Type: Xilinx MicroBlaze Executable	•
✓ ﷺ Debug ✓ ﷺ Release ✓ ﷺ Profile	

DipTest – main.c

- Simple DipTest main.c application created
- Automatic build is set by default (Project → Build Automatically)
- SW platform created (generated from .MSS file)

SW platform

Microblaze_0_sw_platform (right click -> generate Libraries and BSP or LIBGen icon)

- Archives: .a (binary)
- Microblaze_0
 - Code
 - Include*
 - See xparameters.h (generated from .MHS)
 - Lib
 - LibSrc



DipTest SW application

DipTest {microblaze_0_sw_platform}

- Binaries (.elf)
- Debug (.elf)
- main.c
- Additional headers and sources



GPIO drivers and applications

- c:\Xilinx\10.1\EDK\sw\XilinxProcessorIPLib \drivers\
 - gpio_v2_12_a: GPIO v2.12 driver functions
 (low- and high-level driver functions [cpp, h])
 - /Build: OS dependent Makefiles
 - /Data: gpio_header.h + .tcl + .mdd (declares GPIOInput/OutputExample() functions
 - /Doc: API in html form (see index.html)
 - /Examples: simple example applications (use drivers)

- e.g. xgpio_tapp_example.c

• /Src: sources of low-, and higher-level drivers

gpio_header.h

- Declares GpioInputExample() function for prototyping
- XStatus GpioInputExample (Xuint16 DeviceId, Xuint32 *DataRead);
 - #include "xbasic_types.h"
 - #include "xstatus.h"

xgpio_tapp_example.c

- **Declares** GpioInputExample() function
 - This performs a test on the GPIO driver/ device with the GPIO configured as INPUT
 - Invokes low level drivers

XStatus GpioInputExample(Xuint16 DeviceId, Xuint32 *DataRead); /*Function Prototype*/

XGpio GpioInput; /* The driver instance for GPIO Device configured as I/P */

xgpio_tapp_example.c (cont.)

Defines GpioInputExample() **function as follows**:

```
Status GpioInputExample (Xuint16 DeviceId, Xuint32 *DataRead) {
       XStatus Status;
/*
* Initialize the GPIO driver so that it's ready to use,
* specify the device ID that is generated in xparameters.h
*/
       Status = XGpio Initialize(&GpioInput, DeviceId);
       if (Status != XST SUCCESS)
               return XST FAILURE;
       }
/*
* Set the direction for all signals to be inputs
                                                       */
       XGpio SetDataDirection (& GpioInput, GPIO CHANNEL,
  OxFFFFFFF;;
/*
 * Read the state of the data so that it can be verified
                                                               */
       *DataRead = XGpio DiscreteRead(&GpioInput, GPIO CHANNEL);
       return XST SUCCESS;
```

Dependencies

- **#define** GPIO_INPUT_DEVICE_ID XPAR_DIP_DEVICE_ID
 - Cames from xparameters.h (generated from .MHS)
- #define DIP_CHANNEL 1
 - Cames from DIP IP core (remark: GUI settings)
- **#define** GPIO_BITWIDTH 8
 - Cames from 8 DIPSwitches are located on Nexys2 board
- Important note: xil_printf() function must be used instead of normal printf() because the consumes less memory
 - #include stdio.h

Step 2.) Generate Linker Script

- If necessary, set all sections of the .elf file into the internal BRAM memory
 - Select [ilmb_cntlr_dlmb_cntlr] -> Generate

Linker Script Generator							
Application project name: DipTect Heap and Stack:							
· · · · · · · · · · · · · · · · · · ·	Section	Size (bytes)	Memory				
ELF file used to populate section info:	Heap Stack	0x0 0x400	ilmb_cntir_dimb ilmb_cntir_dimb	_cntlr cntlr			
/DipTest/Debug/DipTest.elf							
Code Sections							
Assign all Code Sections to:	Defense						
	Rereren	ice Views (read	-only)				
Section Size (bytes) Memory	Memorie	:S:					
.text 0x00001200 <i>ilmb_cntlr_dlmb_cntlr</i>	Memor	У	A	ddress	Size		
	ilmb_cr	tlr_dlmb_cntlr	0)	<000000000) 32K		
	Micron	RAM_C_MEMO	BASEADDR 0	<8000000C) 16384K		
Add Section Delete Section							
Data Sections	Boot and	d Vector Section	ns:				
	Section	n	Address	Memory			
Assign all Data Sections to:	.vector	rs.reset	0×00000000	ilmb_cntl	lr_dlmb_cntlr	r	
Carther Car Autor) Manager	.vector	rs.sw_exceptio	n 0x00000008	ilmb_cntl	ir_dimb_cntir		
Section Size (bytes) Memory							
.rodata 0x0000042E <i>limb_cntir_dimb_cntir</i>	.vector	S.HW_EXCEPTIO	II 0x00000020	IIIID_cria	r_umb_cria		
data 0v00000134 limb catir dimb catir							
sbss 0x00000000 imp cntir dimp cntir	J						
.bss 0x00000034 ilmb_cntlr_dlmb_cntlr	Output Lin	oker Script:					
	Odtpacia	ikor benper					
D:/FPGA/BEAGYAZOTT_RENDSZEREK/10_1/02_LAB/SDK_projects/I							
1							

SDK: Custom program segments (compile sw application)

- .text the executable code
- .rodata any read-only data used in the execution of the code
- .data where read-write variables and pointers are stored
- .bss a part of the data segment containing statically-allocated variables
- .heap where dynamically allocated memory is located
- .stack where function-CALL parameters and other temporary data is stored

Step 3.) Build SW application

 After building the DipTest sw application the size of the generated, downloadable DipTest.elf file as follows:



Step 4. Terminal Program

- Set the following parameters properly (see the parameters of xps_uartlite in the .mhs file!)
 - Com port: COMX
 - Baud Rate: 9600
 - Data Bits: 8
 - Stop Bits: 1
 - Parity Bit: None
 - Flow control: none

Step 5. Method a.) Programming the FPGA via Xilinx Impact

- Select Device Configuration menu -> Bitstream settings
 - Select compiled DipTest.elf file for running MicroBlaze sw codes

춺 Bitstream Set	tings	2	×				
Specify the ELF file to be marked for BRAM initialization for each processor:							
Processor	Туре	Initialization ELF					
microblaze_0	MICROBLAZE	DipTest/Debug/DipTest.elf 📃 👤					
		Save Cancel	1				

- Connect the Xilinx JTAG-Platform USB cable to Nexys-2 board's JTAG interface
- Select Device Configuration menu -> Program FPGA
 - Bitstream (system.bit)
 - BRAM Memory Map (.bmm) + DipTest.elf
 - --> D:\FPGA\BEAGYAZOTT_RENDSZEREK\10_1\02_LAB\SDK\ SDK_projects\implementation\download_sdk.bit

Step 5. Method b.) Programming the FPGA via Digilent Adept



- Instead of using the Xilinx iMpact, we use
 Digilent Adept Suite! programmer provided by DigilentInc (vendor of the FPGA board).
- Browse your SDK_project\implementation\ directory for "download_sdk.bit" bitstream file.
- Use and set properly the terminal program (e.g. Windows Hyperterminal, Teraterm Pro, or Putty etc.)
- At the final step Program the FPGA!
- At now the Lab 2/A is completed in SDK 10.1
 SP3 [©]

- Open the **system.mhs** file, study its contents, and answer the following questions
- Number of external ports: ______
- Number of external ports that are output (O): _____
- Number of external ports that are input (I): _____
- Num. of external ports that are bidirectional (IO): ____
- Number of clock ports: _____ Freq: _____
- Number of reset ports: _____ Polarity: _____



Lab2 Intro

• List the **instances** to which the clk_s is connected:

• List the **instances** connected to the mb_plb bus:



Lab2 Intro © 2011 Xilinx, Inc. All Rights Reserved

For Academic Use Only

• Draw the **address map** of the system, providing instance names:



For Academic Use Only

© 2011 Xilinx, Inc. All Rights Reserved

Check Report files (system.par) or log messages in Consol window after the placement process step):

Logic Utilization:			
Number of Slice Flip Flops:	out of	17 , 344	119
Number of dipput IUTs.	out of	17 3//	1 0 9
	Out OI	I/, J44	т 9-с
Logic Distribution:			
Number of occupied Slis:	out of	8,672	298
Number of External IOBs	out of 250	22%	
Number of External Input IOBs			
Number of External Output IOBs			
Number of External Bidir IOBs			
Number of BSCANs out of 1 100%			
Number of BUFGMUXs out of 24 8	00		
Number of DCMs Out of 8 12%			
Number of MULT18X18SIOs out of 28	10%		
Number of RAMB16s out of 28 71%			
Number of Slices out of 8672 209			
Number of SIICes Out of 0072 29%			
NUMBER OF SLICEMS OUT OF 4336 68			

•



© 2011 Xilinx, Inc. All Rights Reserved

Lab2 Intro

For Academic Use Only

Laboratory task 2/B: Push Buttons

• In XPS/EDK:

- Similar to the Task 2/A add Push Buttons (4 bit) as new peripheral into the elaborated embedded system design
- Rename it: "push", and add a particular address
- Generate netlist and bitstream
- In XPS/SDK:
 - Similar to the Task 2/A, create modify the previous DipTest sw application in SDK
 - Create PeripheralTestsApp_bsp
 - Implement BSP, generate Linker Script (.ld)
 - Compile MB codes (.elf file)
 - Generate Bitstream (.bit)
- Download bitstream (.bit), and analyze the desing on the Digilent Nexys2 FPGA board.
- Try to answer the questions according to the TASK 2/A.



- What is the size of **.elf** program, and the different program sections?
- Which is the base_address and high_address (or address size) of the push button GPIO peripheral?
- Which header .h file contains the MicroBlaze system parameters for various peripherals?



Lab2 Intro © 2011 Xilinx, Inc. All Rights Reserved

For Academic Use Only