### **Experiments: Some Preparations ....**

### 1) VITIS application templates

There are template for VITIS projects at:

# Windows:?:\Xilinx\Vitis\2022.1\data\embeddedsw\lib\sw\_appsLinux:/scratch/Xilinx/Vitis/2022.1/data/embeddedsw/lib/sw\_apps

### copy: *HrFreertos\_HelloWorld HrFreertos\_lwip\_echo* into this directory.

### 2) IP Block for PMOD\_AD1 copy: IP\_Pmod\_AD1

into your Vivado Project directory

ame	Änderungsdatum	Тур	Größe
treertos_hello_world	07.11.2022 19:34	Dateiordner	
freertos_lwip_echo_server	07.11.2022 19:34	Dateiordner	
freertos_lwip_tcp_perf_client	07.11.2022 19:33	Dateiordner	
freertos_lwip_tcp_perf_server	07.11.2022 19:18	Dateiordner	
freertos_lwip_udp_perf_client	07.11.2022 19:33	Dateiordner	
freertos_lwip_udp_perf_server	07.11.2022 19:18	Dateiordner	
hello_world	07.11.2022 19:26	Dateiordner	
HrFreertos_HelloWorld	14.11.2022 17:40	Dateiordner	
HrFreertos_Iwip_echo	09.11.2022 08:29	Dateiordner	
img_rcvry	07.11.2022 19:41	Dateiordner	
imgsel	07.11.2022 19:41	Dateiordner	
libmetal_echo_demo	07.11.2022 19:33	Dateiordner	
lwip_echo_server	07.11.2022 19:33	Dateiordner	
lwip_tcp_perf_client	07.11.2022 19:18	Dateiordner	
lwip_tcp_perf_server	07.11.2022 19:18	Dateiordner	
lwip_udp_perf_client	07.11.2022 19:33	Dateiordner	
lwip udp perf server	07.11.2022 19:33	Dateiordner	

### **Experiment: List of Material**

#### ZedBoard

Q Ship					ANMELDE	n Qwisheist	T
Shep All	Boards and Components	Test and Measurement Equipment	<b>Bundles and Specials</b>	Software	Academic	Applications	Support
		BBBB US shippi					

Startseller / System Roads and Components / Judicard Jysg-2000 ARM/IPGA Soc Development Road



#### ZedBoard Zynq-7000 ARM/FPGA SoC Development Board \$589.00 \*\*\*\*\* Offerensinged Breesing schroben

50.0 400-248

As low as \$54/mo with affirm Loarn more

Add ens: (Driorderlich) Zeddoard Zyng 7000 Add 505oC Voucher

Hange:

### or alternative **ZYBO**



Startselte / System Boards and Components / Zybo Zh Zyng-Jtoti ARM/FPGA Soc Dewlopment Board



#### Zybo Z7: Zynq-7000 ARM/FPGA SoC Development Board \$299.00

\*\*\*\*\*

90,1 410-351-10

#### PMOD AD1



# Prod AD1: Two 12-bit A/D Inputs Currently out of stock. Estimated return is June 2023. R # # R (back here hereinagen) thereinage is back. Still e 0 del Over Stellador Stock Date Stellador Stock Date Stellador Stock Date Stellador Stock Date Stellador Stock

Teally No.

### **Camera Module**



### **Generic Experiment Setup**



- VHDL code

- $\mu C$  code for DAQ
- and Ethernet communication
- Visualization

### ZedBoard & PMOD extensions







#### Pmod Expansion on ZedBoard



Pmod	Signal Name	Zynq pin	Pmod	Signal Name	Zynq pin
	JA1	Y11		JB1	W12
JA1	JA2	AA11	JB1	JB2	W11
	JA3	Y10		JB3	V10
	JA4	AA9		JB4	W8
	JA7	AB11		JB7	V12
	JA8	AB10		JB8	W10
	JA9	AB9		JB9	V9
	JA10	AA8		JB10	V8

Pmod	Signal Name	Zynq pin	Pmod	Signal Name	Zynq pin
JC1 Differential	JC1_N	AB6		JD1_N	W7
	JC1_P	AB7		JD1_P	V7
	JC2_N	AA4	JD1	JD2_N	V4
	JC2_P	Y4		JD2_P	V5
	JC3_N	T6	Differential	JD3_N	W5
	JC3_P	R6		JD3_P	W6
	JC4_N	U4		JD4_N	U5
	JC4_P	T4		JD4_P	U6

Pmod	Signal Name	Zynq pin
	JE1	A6
	JE2	G7
JE1 MIO Pmod	JE3	B4
	JE4	C5
	JE7	G6
	JE8	C4
	JE9	B6
	JE10	E6





### PMOD AD1 (2 channel, 12 bit, 1 MSPS)

Features include:

- two AD7476A 12-bit A/D converter chips
- a 6-pin header connector
- a 6-pin connector
- two 2-pole Sallen-Key anti-alias filters
- two simultaneous A/D conversion channels at up to one MSa per channel
- very low power consumption
- small form factor (0.95" x 0.80").







Pmod	Signal Name	Zynq pin	
	JB1	W12	CS
	JB2	W11	Data_1
	JB3	V10	Data 2
ID1	JB4	W8	SCLK
JDT	JB7	V12	
	JB8	W10	
	JB9	V9	
	JB10	V8	

### **Experiment 1: First freeRTOS application**



#### **PMOD AD Interface**

- Create IP Block
- VHDL code

# Create SOC Hardware SDK: freeRTOS

- μC code for DAQ

#### **Debug Terminal**

### **VIVADO:** Create Hardware Platform for Experiment



#### Constraints:

#PMOD AD1 on JB1
set\_property IOSTANDARD LVCMOS33 [get\_ports {SPI\_AD1\_\*} ]
set\_property PACKAGE\_PIN W11 [get\_ports SPI\_AD1\_io0\_i]
set\_property PACKAGE\_PIN W10 [get\_ports SPI\_AD1\_io1\_i]
set\_property PACKAGE\_PIN W12 [get\_ports SPI\_AD1\_ss1\_0]
set property PACKAGE PIN W8 [get ports SPI AD1 sck o]

### **VITIS: Create Application Project**

- 1) Create new Application Project
  - Platform: create from exported hardware \*.XSA

Create a	new plat	form from hardware ()	(SA)	Select a platform from repository	
Hardware	e Specific	ation			
	Provide	your XSA file or use a pr	e-bu	ilt board description	
	vck190 vmk180	Vereate Platform fro	m XS	5A	
XSA File:	zc702 zc706 zcu102	← → • ↑ 🖡	> D	ieser PC → Daten (D:) → Trainings → ict	p22_1
	zcu106 zed	Organisieren 🝷 🛛 I	Veue	r Ordner	
			^	Name	
		🗸 🧚 Schnellzugriff		FW	
Software	Specifica o dotaile	Desktop	*	ictp22 1.cache	
Operation		👆 Downloads 🗧	*	ictp22_1.gen	
Operating	y system.	💄 work 👘	*	ictp22_1.hw	
Processo		on 1_Waldfeucht	*	ictp22_1.ip_user_files	
		> 🐼 sciebo		ictp22_1.sim	
				ictp22_1.srcs	
		🗸 🧢 Dieser PC		IP_Pmod_AD1	
		> 📃 Desktop		design_1_wrapper.xsa	
		Devuele ede			

### **VITIS: Create Application Project**

#### 2) Application Project ۲

New Application project:

Operating System:

freertos10\_Xilinx

Select template \_\_\_\_

#### 'HR FreeRTOS IwIP echo server'

🖌 New Application Project		
Domain		
Select a domain for your project or create a new de	omain	
Select the domain that the application would link t Note: New domain created by this wizard will have	to or create a new dor e all the requirements	main of the application template selected in the next st
Select a domain	Domain details	
+Create new	Name:	freertos10_xilinx_ps7_cortexa9_0
	Display Name:	freertos10_xilinx_ps7_cortexa9_0
	Operating System:	freertos10_xilinx
	Processor:	ps7_cortexa9_0
	Architecture:	32-bit ~ 4
		F

#### New Application Project

#### emplates

ig
angle This application requires IwIP library in the Board Support Package. You can go back to the previous page platform and domain or create a new one with a suitable hardware and software.

#### vailable Templates:

<ul> <li>Embedded software development templates</li> </ul>	HR FreeRTOS IwIP application
Empty Application (C++)	
Empty Application(C)	
FreeRTOS Hello World	
FreeRTOS IwIP Echo Server	
FreeRTOS IWIP TCP Perf Client	
FreeRTOS IWIP TCP Perf Server	
FreeRTOS IWIP UDP Perf Client	
FreeRTOS IWIP UDP Perf Server	
HR FreeRTOS Hello World	
HR FreeRTOS IwIP Echo Server	
OpenAMP echo-test	
OpenAMP matrix multiplication Demo	
OpenAMP RPC Demo	

### VITIS:

- Create a project based "Hr FreeRTOS lwIP echo server"
- Compile, make Run Configuration, download to Zedboard
- Connect a terminal to the Serial line Debug output (Teraterm, ...)
- Check for "Hello World"
- Add a task for controlling the LEDs (via GPIO)
- Add a task for readout of the ADC
  - Print the ADC value

### Experiment 2: Realtime Data Acquisition System based on freeRTOS & IwIP



Ethernet TCP/IP

MT niScope Demodulate AM.vi File Edit Operate Tools Browse Window

**PMOD AD Interface** 

- Create IP Block
- VHDL code

Create SOC Hardware SDK: freeRTOS + IwIP

- μC code for DAQ
- and Ethernet communication

**Network software** 

- Receive Data
- Visualization

### VITIS:

- Enable 'task\_StartEthernet' in main.c
- Setup the PC network to match the Zedboard IP address space (192.168.2.2)
- Check the debug serial for established Ethernet connection
- On the PC:
  - Use 'ping' for checking the Ethernet connection
  - Start a Brower and connect to 192.168.2.2
- Modify the 'webserver' to enable the ADC
  - Print the ADC value in html language
  - Check with the browser
  - Enable the html code for the visualization of the graph
  - Check with the browser

### Appendix

### **Labview Client for Visualization**



### **AD7476 serial ADC: Timing**

- Cycle time: 1 µs (=1 MHz Sample rate)
  - AXI Clock (100 MHz) divide by 100
- SCLK
  - Max. 20 MHz = AXI Clock / 5
  - Realized in a Statemachine of 5 states
- 0) SCLK =1 / CS = 0
- 1) SCLK =1
- 2) SCLK =0 / Read serial Bit
- 3) SCLK =0 / increment Bit nr
- 4) SCLK =0 / check for 16 bits Jump to 0 or IDLE



### **VHDL** sniplet

use ieee.std\_logic\_unsigned.all;

port	( Users	to add ports here
	CS	: out std_logic;
	SCLK	: out std_logic;
	SDATA1	: in std_logic;
	SDATA2	: in std_logic;
•••		

signal	type signal Pacer1M	TSTATES is sState : std_logic;	(IDLE, S0, S1, S2, S3, S4 ); : TSTATES;
signal	AdcDatal	: std_logic_v	vector (15 downto 0);
signal	AdcData2	: std_logic_v	vector (15 downto 0);

```
pAD7576t: process (S_AXI_ACLK)
                       variable BitNr : integer range 0 to 31;
                        variable InDat1, InDat2 : std_logic_vector (15 downto 0);
           begin
           if rising_edge (S_AXI_ACLK) then
            case sState is
                        when IDLE =>
                                            <= '1';
                                       SCLK
                                       CS <= '1';
                                       BitNr := 0;
                                       if (Pacer1M = '1') then
                                       CS <= '0';
                                         sState <= S0;
                                       end if;
                        when SO
                                => SCLK <= '1';
                                       sState <= S1;
                        when S1 => SCLK <= '1';
                                       sState <= S2;</pre>
                        when S2
                                  => SCLK <= '0';
                                       InDat1 := InDat1 (14 downto 0) & SDATA1;
                                                                                   -- serial SHIFT IN
                                       InDat2 := InDat2 (14 downto 0) & SDATA2;
                                                                                      -- serial SHIFT IN
                                       BitNr := BitNr +1;
                                       sState <= S3;</pre>
                        when S3
                                  => SCLK
                                            <= '0';
                                       sState <= S4;
                        when S4
                                 => SCLK <= '1';
                                       if (BitNr < 16) then
                                                   sState <= S0;</pre>
                                       else
                                                  CS <= '1';
                                                  AdcData1 <= InDat1;
                                                  AdcData2 <= InDat2;
                                                  sState <= IDLE;</pre>
                                       end if;
            end case;
            end if;
           end process;
```

### **Pin Constraints**

#--- pinning for Pmod Connector B (upper level) ------

set\_property PACKAGE\_PIN W12 [get\_ports AD1\_CS]
set\_property PACKAGE\_PIN W11 [get\_ports AD1\_SDATA1]
set\_property PACKAGE\_PIN V10 [get\_ports AD1\_SDATA2]
set\_property PACKAGE\_PIN W8 [get\_ports AD1\_SCLK]

set\_property IOSTANDARD LVCMOS33 [get\_ports AD1\_CS]
set\_property IOSTANDARD LVCMOS33 [get\_ports AD1\_SCLK]
set\_property IOSTANDARD LVCMOS33 [get\_ports AD1\_SDATA1]
set\_property IOSTANDARD LVCMOS33 [get\_ports AD1\_SDATA2]

- Now Generate BitFile
- Export
- Launch SDK

### Application based on ,hrfreertos\_lwip\_web\_tcp" Changes in ,,Echo.c"

```
void ReadADC_Fast (void)
{
int i;
  for (i=0; i<1024; i++)</pre>
  {
     AdData[i] = (short) *(int*)0x43C10000;
                                                                  // direct IO access
     usleep(2);
  }
}
void process_echo_request(void *p)
{
....
while (1)
{
...
           /* handle request */
          ReadADC_Fast ();
          wrote = write(sd, AdData, 2048);
           {
•••
}
```

short AdData[1024];

### **Annex: Accessing memory mapped Hardware-Devices**

**Using BSP driver functions:** #include "xparameters.h" #include "xgpio.h" #define LED CHANNEL 1 2 #define SW CHANNEL XGpio Gpio; /\* The Instance of the GPIO Driver \*/ // GPIO Initialisation: XGpio\_Initialize (&Gpio, XPAR\_AXI\_GPIO\_0\_DEVICE\_ID); //=0 XGpio\_SetDataDirection (&Gpio, LED\_CHANNEL, 0xFFFFFF00); // 0 = OutputsXGpio\_SetDataDirection (&Gpio, SW\_CHANNEL, // 1 = Inputs 0xffffffff); //GPIO Data: Data = XGpio\_DiscreteRead (&Gpio, SW\_CHANNEL); XGpio\_DiscreteWrite (&Gpio, LED\_CHANNEL, Data);

#### Using direct I/O functions:

#include "xparameters.h"
#include "Xil\_io.h"

Xil\_Out32 (Addr, Value)
Xil\_In32 (Addr)

#### **Pointer usage:**

### **On Stick**



- Some books..
- VHDL IP Blocks for Vivado IP Integrator
- The PMOD documentation
- The Software Repository (freeRTOs & IwIP)
- A Terminal
- Some tools
- Zedboard Docu

This file ...!

٠

### **Pulse Oximeter**

### **Heart Beat Monitor**

Dr. Heinz Rongen

### **Pulse Oximeter**

- medical device that indirectly monitors
  - the oxygen saturation of a patient's blood
  - and changes in blood volume in the skin
- $\rightarrow$  Monitoring of heart rate
- Infrared LED (900 to 950 nm)
- Absorption at these wavelengths differs significantly
  - between oxyhemoglobin and its deoxygenated form
  - the arterial expand and contract with each heartbeat



#### **TCRT1000, TCRT1010**

**Vishay Semiconductors** 

#### **Reflective Optical Sensor with Transistor Output**



#### DESCRIPTION

The TCRT1000 and TCRT1010 are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light.

#### FEATURES

- Package type: leaded
- Detector type: phototransistor
- Dimensions (L x W x H in mm): 7 x 4 x 2.5
- · Peak operating distance: 1 mm
- RoHS • Operating range within > 20 % relative collector current: 0.2 mm to 4 mm
- Typical output current under test: I<sub>C</sub> = 0.5 mA
- Daylight blocking filter
- · Emitter wavelength: 950 nm
- Lead (Pb)-free soldering released
- · Compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC

#### APPLICATIONS

 Optoelectronic scanning and switching devices i.e., index sensing, coded disk scanning etc. (optoelectronic encoder assemblies for transmissive sensing).





IR-LED and IR-Detector as reflective Sensor

دددد 

high absorption





COMPLIANT

**TCRT1010** 



### TCRT1000, TCRT1010

Vishay Semiconductors

### **Reflective Optical Sensor with Transistor Output**



#### DESCRIPTION

The TCRT1000 and TCRT1010 are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light.

#### FEATURES

- · Package type: leaded
- Detector type: phototransistor
- Dimensions (L x W x H in mm): 7 x 4 x 2.5
- Peak operating distance: 1 mm
- Operating range within > 20 % relative collector current: 0.2 mm to 4 mm
- Typical output current under test: I<sub>C</sub> = 0.5 mA
- · Daylight blocking filter
- Emitter wavelength: 950 nm
- · Lead (Pb)-free soldering released
- Compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC

#### APPLICATIONS

• Optoelectronic scanning and switching devices i.e., index sensing, coded disk scanning etc. (optoelectronic encoder assemblies for transmissive sensing).





ROHS COMPLIANT

### **Preamplifier (with Filter)**



### ZYNQ Stand-Alone Booting (SD-Card or QSPI Memory)



### Programming the ZedBoard

- USB-JTAG Default method for prototyping
- **Traditional JTAG** A Xilinx JTAG connector. Requires a *Xilinx Platform USB* cable

• **SD card** — can be used to program the ZYNQ with files stored on the SD card.

• Quad-SPI flash memory — Non-volatile flash-memory.



Boot mode latched at POR using MIO pins by BootROM

Mode	MIO[5] (JP10)	MIO[4] (JP9)	MIO[3] (JP8)
JTAG	0	0	0
NOR	0	0	1
NAND	0	1	0
Quad-SPI	1	0	0
SD Card	1	1	0

### Prepare for a SD card Image

### Create a First Step Boot Loader (FSBL)

• File  $\rightarrow$  New  $\rightarrow$  Application

New Project – 🗆 🗙	
Application Project	
A project with that name already exists in the workspace.	<ul> <li>Name is "** FSBL"</li> </ul>
Project name: myFSBL	—
✓ Use default location	
Location: D:\Rio\1_FirstSOC\1_FirstSOC.sdk\myFSBL Browse	
Choose file system: default 🗸	
Target Hardware	
Hardware Platform: design_1_wrapper_hw_platform_0 V New	
Processor: ps7_cortexa9_0 v	
Target Software	
Language:	• Lot him croate a new DCD or the FCDI
OS Platform: v	• Let min create a new BSP of the FSBL
Board Support Package:  Create New myFSBL_bsp	
○ Use existing myFSBL_bsp ∨	
	Templates
	Create one of the available templa
	project.
Omega          And Cancel           Image: Concel         Image: Concel         Image: Concel	Available Templates:
	Peripheral Tests Dhrvstone
Click NF	Empty Application
	Hello World
and sel	ect the "7vna FSBI"
	RSA Authentication App
	Xilkernel POSIX Threads Demo
The FCDI is now	Zynq DRAM tests
I NE FORL IS NOW	

### Create a Zynq Boot Image for SD Card

### Xilinx Tools $\rightarrow$ Create Zynq Boot Image



SOK	Create Zyr	nq Bo	ot Im	age			×
Create Zynq Boot Creates Zynq Boot I	: Image mage in .bin and .mcs formats from given FSBL elf a	and par	tition	files in specified	output folder	:	-co
Create new BIE file	Import from existing BIF file						
Import BIF file path:	D:\Rio\1_FirstSOC\1_FirstSOC.sdk\MyFirstSocAppl\	bootim	age\N	lyFirstSocAppl.b	if		Browse
Output BIF file path:	: D:\Rio\1_FirstSOC\1_FirstSOC.sdk\MyFirstSocAppl\bootimage\MyFirstSocAppl.bif						
Use Authentication	1						
Authentication keys							
PPK:	E	Browse	PSK:				Browse
SPK:	B	Browse	SSK:				Browse
SPK signature:	E	Browse					
Use encryption							
Encryption key:							
Key file:							Browse
Key store: 🖲 BRAN	1 O EFUSE						
Part name:							
Boot image partitions	;						
File path			I	Encrypted	Authentic		Add
(bootloader) D:\Rio\1_FirstSOC\1_FirstSOC.sdk\myFSBL\Debug\myFSBL.elf		.elf		none	none		Delete
D:\Rio\1_FirstSOC\1_FirstSOC.sdk\MyFirstSocAppl\Debug\MyFirstSocAppl.elf		pl.elf		none	none		Edit
							Up
							Down
Output path: D:\Ric	\1_FirstSOC\1_FirstSOC.sdk\MyFirstSocAppl\bootim	nage\B(	DOT.bi	n			Browse
?			Previe	ew BIF Changes	Create I	mage	Cancel

### Add the following files:

- the FSBL
- the BIT file
- the ELF file
- Click "Create Image"
- BOOT.BIN is created

### **Directories to find the components**

Create Zyng Boot Image in SDK

	🚱 Create Zynq Boot Image			×					
Graphical     front and to	Create Zynq Boot Image Creates Zynq Boot Image in .bin and .mc	Create Zynq Boot Image Creates Zynq Boot Image in .bin and .mcs formats from given FSBL elf and partition files in specified output folder.							
front end to	Create new BIF file Import from exis	Create new BJF file							
command	BIF file path C:\Speedway\ZynqSW\2013_3\SDK_Workspace\Test_Peripherals\bootimage\Test_Peripherals.bif								
line bootgen	Use Authentication Authentication keys								
	ррк	Browse		Browse					
	SPK	Browse		Browse					
	SPK Signature	Browse							
Partitions	Part name	Part name							
1. FSBL	File path		Encrypted Authenticat	d Add					
<ol> <li>2. Bitstream</li> <li>3. Application</li> </ol>	(bootloader) C:\Speedway\ZynqSW\2013 C:\Speedway\ZynqSW\2013_3\SDK_Work C:\Speedway\ZynqSW\2013_3\SDK_Work	_3\SDK_Workspace\zynq_fsbl_0\Release\zynq_fsbl_0.elf space\ZynqHW\Z_system_wrapper.bit space\Test_Peripherals\Release\Test_Peripherals.elf	none none none none none none						
				Edit Down					
		m		Edit Up Dowr					
	C:\Speedway\ZynqSW\20	m 113_3\SDK_Workspace\Test_Peripherals\bootimage\outpu	ıt.bin	Edit Up Down					

- First Step Boot Loader
   \*.sdk\myFSBL\Debug
- FPGA configuration \*.runs\impl\_1
- Application
  - \*.sdk\MyFirstSocAppl\Debug
  - Is the order of the boot images critical? If so, list the order.

YES! FSBL ELF, then bitstream, then application ELF



## • all in **BOOT.BIN**



### **Booting from SD Card**

- Copy BOOT.BIN on a SD card
- Set Jumper according
- Insert CD card
- Power Cycle the ZedBoard



### **Booting from FLASH (QSPI Memory)**

### • Xilinx Tools $\rightarrow$ Program Flash



- Browse the BOOT.BIN
- $\rightarrow$  Program
- Needs some time ...



### **Booting from FLASH (QSPI Memory)**

. Turn off the ZedBoard. Verify the Configuration Mode jumpers are set for QSPI boot mode as described and in the figure below:

- MODE3 (JP10) shunted to 3.3V
- All other MODE pins shunted to GND



### **Generic Experiment Setup**





#### **PMOD AD1 Interface**

- Create IP Block
- VHDL code

Vivado: Create SOC Hardware SDK: freeRTOS based Application - μC code for DAQ

### **Demonstration 1**





#### PMOD AD1 Interface

- Create IP Block
- VHDL code

Vivado: Create SOC Hardware SDK: freeRTOS based Application - μC code for DAQ