# 芯动力——硬件加速设计方法

# **第七章 基于平头哥E902处理器的SoC设计** —(5) 基于NexysVideo板卡的FPGA实现和I/O LAB实验

邸志雄@西南交通大学 zxdi@home.swjtu.edu.cn

slides与源代码网址 http://www.dizhixiong.cn/class5/

本课程SOC教学中所有实验内容都使用Nexys Video板卡完成。 只要FPGA板卡的资源能够承载无剑SoC,都 可以用来学习与实践本课程的内容。 IO LAB实验

- ◎ 串口
- OLED
- LED
- ◎ SD卡读写
- ◎ 按键中断驱动





NexysVideo是Digilent 公司专为音视频应用设计的 板卡,搭载了Xilinx® Artix-7系列中功能最强大的芯片 Artix®-7 FPGA XC7A200T,资源容量非常大。









Digilent官方有该FPGA板卡的板级支持文件,在建立工程时即可选择板卡配置而无需对具体的 FPGA芯片型号进行选择。除此之外还提供了每个板卡的xdc约束脚本文件,便于进行信号与引脚的 配置。上述文件Digilent均在github上开源。

🗆 🚽 🕆 📍 🔍 🚽 🚽 🔿 🚽 🚽	do2018.3 > Vivado > 2018.3 > data	a > boards > board_files	Parts Boards Reset All Filters	
<sup>称</sup> 安装	在viva的动力应断	5本散指完位 響	Vendor: All Vame: All	
kintex7	2021/5/3 10:28		Search: Q∴nexys vi ⊗ ♥ ( Display Name	I match) Preview Vendor File Version Part
nexys_video	2021/5/3 13:07	文件夹	Nexys Video	digilentinc.com 1.1 xc7a200tsbg48
nexys4	2021/5/3 13:07	文件夹		
nexys4_ddr	2021/5/3 13:07	文件夹		
nexys-a7-50t	2021/5/3 13:07	文件夹		
nexys-a7-100t	2021/5/3 13:07	文件夹		
nico mEOE	2021/5/2 10:28	文件范		



## ②添加设计源文件

wujian100 开源工程





通过Synplify对SoC进行综合,然后把其综合结果edf网表文件在Vivado上进行布局布线,最终生成bitstream文件。使用Synplify的原因在于Synplify是功能超强的FPGA综合软件。是业界领先的FPGA逻辑综合解决方案。

+



#### #project files

add\_file -verilog ".//wujian100\_open\_fpga\_top.v" add\_file -verilog "././soc/ahb\_matrix\_top.v" add\_file -verilog "././soc/smu\_top.v" add\_file -verilog "././soc/sms.v" add\_file -verilog "././soc/ls\_sub\_top.v" add\_file -verilog "././soc/retu\_top.v" add\_file -verilog "././soc/rtim5.v" add\_file -verilog "././soc/tim5.v" add\_file -verilog "././soc/tim.v" add\_file -verilog "././soc/dmac.v" add\_file -verilog "././soc/pdu\_top.v"

set\_option -project\_relative\_includes 1
set\_option -include\_path {../../soc/params/}
add file -constraint "wujian100 open.sdc"

### 综合时需要添加的设计文件(Synplify工程文件)







- ◎ 时钟信号
- 复位信号
- JTAG调试信号
- ◎ 串口
  - 这些约束最终需要放在xdc约束脚本 中与设计源文件一起综合实现。





### 这些约束最终需要放在xdc约束脚本中与设计源文件一起综合实现。



以GPIO\_0为例,在xdc约束脚本中将其在SoC设计的信号分配至硬件上与LED相连接的FPGA引脚,再 通过软件中的GPIO PA0完成对GPIO\_0的控制,从而实现LED亮灭的控制。硬件实现和软件控制上要保 证一致性。

# FPGA实现流程 ③板卡适配:时钟信号

## 时钟信号的分配

- 板载晶振为100MHz
- ▶ wujian100 SoC在Nexy Video板卡的FPGA型号上工作最高主频为20MHz

这里需要采用Vivado clocking wizard IP进行分频,再将分频后20MHz的时钟信号送入PIN\_EHS 信号提供给SoC。该分频模块的复位信号由系统复位输入提供,低电平复位。





注意

## ③板卡适配:时钟信号

时钟信号的产生需要使用时钟分频模块,一定要用clocking wizard IP,不要使用自己用counter实现的时钟分频模块。

FPGA内部全局时钟布线资源

逻辑布线资源

时钟布线资源较少但是在整个芯片上传输非常快、相位延迟也很低,输出信号和输入时钟有固定的相位关系;

逻辑资源分布在整个芯片上,延时大、时钟歪斜严重,并且在综合实现过程中,因优化程度不同每次时序关系都会发生改变。





SoC共集成有三个USI模块,其中USI0模块默认被用作串口并完成了printf函数的串口重定向,可以通过使用printf函数直接通过串口将内容输出。

FT2232接口转换芯片主要功能

①为FPGA加载bitstream

②USB转换成UART与FPGA引脚相连。



Serial Ports(PAD)	UART	I2C-	SPI
SCLK←	RXD↩	SCL↩コ	SCK↩
SD0← <sup>□</sup>	TXD↩	SDA←	MOSI∈
SD1↩	CTS↩	$\leftarrow$	MISO∉
NSS↩	RTS↩□	€ <sup>2</sup>	NSS↩

## UART

set\_property -dict { PACKAGE\_PIN AA19 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_USI0\_SD0 }]; #IO\_L15P\_T2\_DQS\_RDWR\_B\_14 Sch=uart\_rx\_out set\_property -dict { PACKAGE\_PIN V18 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_USI0\_SCLK }]; #IO\_L14P\_T2\_SRCC\_14 Sch=uart\_tx\_in

硬件设计



③板卡适配:串口信号



Serial Ports(PAD)	UART	I2C-	SPI
SCLK←	RXD↩	SCL↩コ	SCK↩
SD0↩	TXD↩	SDA←	MOSI↩
SD1↩	CTS↩	$\leftarrow$	MISO⋳
NSS↩	RTS↩□	Ę	NSS↩

## UART

set\_property -dict { PACKAGE\_PIN\_AA19 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_USI0\_SD0 }]; #IO\_L15P\_T2\_DQS\_RDWR\_B\_14 Sch=uart\_rx\_out set\_property -dict { PACKAGE\_PIN\_V18 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_USI0\_SCLK }]; #IO\_L14P\_T2\_SRCC\_14 Sch=uart\_tx\_in

硬件设计



这里将USIO的串口模块的RXD/TXD引脚分配至FPGA的V18/AA19后,即可在SDK直接使用printf相关函数,便于调试。

### 本课程SOC教学中所有实验内容 都使用Nexys Video板卡完成

### 只要FPGA板卡的资源能够承载无 剑SoC,都可以用来学习与实践 本课程的内容。

## I/O LAB实验

- ◎ 串口
- OLED
- LED
- ◎ SD卡读写
- ◎ 按键中断驱动





在C语言标准库中,fputc函数是printf函数内部的一个函数,功能是将字符写入到文件指针所指 向文件的当前写指针位置,简单理解就是把字符写入到特定文件中。

通过将串口发送函数重新修改 fputc函数内容,达到运行printf函数时从串口输出字符串。同样,通过将串口接收函数重新修改fgetc函数即可调用scanf、getchar等函数,通过串口获取字符。

```
int fputc(int ch, FILE *stream)
                                                                 int fgetc(FILE *stream)
   (void)stream;
                                                                     uint8_t ch;
   if (console_handle == NULL) {
                                                                     (void)stream;
       return -1;
                                                                     if (console_handle == NULL) {
                                                                         return -1;
   if (ch == '\n') {
       csi_usart_putchar(console_handle, '\r');
                                                                     csi_usart_getchar(console_handle, &ch);
   csi usart putchar(console handle, ch);
                                                                     return ch;
   return 0;
                                     重定向printf、scanf函数到串口
```



return 0;

## ③板卡适配:串口信号

wujian100 SDK中在进入main用户应用程序前会先将USI0模块初始化并配置为串口通信方式, 且波特率为115200。SDK还完成了printf、scanf函数的重定向,因此用户可以直接在应用程序中 调用printf、scanf相关函数,完成串口的输入输出,便于用户调试。

```
int fputc(int ch, FILE *stream)
{
    (void)stream;
    if (console_handle == NULL) {
        return -1;
     }
    if (ch == '\n') {
        csi_usart_putchar(console_handle, '\r');
     }
    csi_usart_putchar(console_handle, ch);
```

```
int fgetc(FILE *stream)
{
    uint8_t ch;
    (void)stream;
    if (console_handle == NULL) {
        return -1;
     }
    csi_usart_getchar(console_handle, &ch);
    return ch;
}
```

重定向printf、scanf函数到串口

# FPGA实现流程 ③板卡适配:复位信号、调试信号



wujian100 SoC中复位信号为PAD\_MCURST,低电平复位,Clocking Wizard IP的复位信号也设置为此信号提供,这里将其分配至按键上,当按键按下后处理器复位。



wujian100 SoC采用双线JTAG接口,这里将其分配至PMOD接口上,便于 与调试器连接。Nexys Video板卡上共有3个PMOD接口,可任意选择一个 进行分配。

为了在后续I/O LAB实验中进行更多 的应用,这里还需要把wujian100 SoC的相关信号分配至LED/switch 开关和OLED等。





复位按键和PMOD接口

PMOD接口

### FPGA实现流程 ③板卡适配:LED控制信号

将wujian100 SoC中GPIO控制相关信号PAD\_GPIO\_X分配至FPGA与LED灯相连的引脚,当软件 控制GPIO输出高电平时LED灯点亮,输出低电平时LED熄灭。



set\_property -dict { PACKAGE\_PIN T15 IOSTANDARD LVCMOS25 } [get\_ports { PAD\_GPIO\_0 }]; = set\_property -dict { PACKAGE\_PIN T16 IOSTANDARD LVCMOS25 } [get\_ports { PAD\_GPIO\_1 }]; : GND set\_property -dict { PACKAGE\_PIN U16 IOSTANDARD LVCMOS25 } [get\_ports { PAD\_GPIO\_2 }]; set property -dict { PACKAGE PIN V15 IOSTANDARD LVCMOS25 } [get ports { PAD GPIO 3 }]; 硬件设计

软件实现

while (1) {

} else {

为实现LED灯每秒闪烁一次,在程序中使用提供的mdelay函数实 现, 该函数利用处理器内部的定时器实现较准确的ms级延迟。





ms级延时的实现

### FPGA实现流程 ③板卡适配:OLED显示控制信号

OLED显示器与LED点阵彩色显示器的原理类似,但它采用的像素单元是"有机发光二极管",所以 像素密度比普通LED点阵显示器高得多。OLED显示器具有不需要背光源、对比度高、轻薄、视角广 及响应速度快等优点。

Nexys Video板卡上搭载有128\*32的OLED显示器和SSD1306显示驱动芯片,其面向FPGA的为四线SPI接口,通过SPI协议配置该芯片或传输数据从而完成显示器的控制。此处通过GPIO实现 软件SPI协议控制显示驱动芯片。

Signal	Description	Polarity	FPGA pin	V(C2)V2 2 14 2
RES#	Reset	Active-low	U21	
CS#	Chip select (always active)	Active-low	N/A	OLED_VBAT
D/C#	Data (high)/Command (low)	Both	W22	VCC3V3 2 3
SCLK	Serial Clock	Active-high	W21	
SDIN	Serial Data	Active-high	Y22	OLED_VDD
VBAT#	Power enable for internal power supply	Active-low	P20	
VDD#	Power enable for digital power	Active-low	V22	

#### Table 112. OLED signal description.

## OLED Display

set property -dict { PACKAGE PIN W22 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 21 }] set property -dict { PACKAGE PIN U21 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 22 }]; set property -dict { PACKAGE PIN W21 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 23 }] set\_property -dict { PACKAGE\_PIN Y22 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_GPIO\_24 }]; set property -dict { PACKAGE PIN P20 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 25 }]; set\_property -dict { PACKAGE\_PIN V22 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_GPIO\_26 }]; 硬件设计



OLED显示驱动的SPI控制协议和实现如下图。在CS片选信号低电平时开始进行数据传输,在SCLK时钟的上升沿完成数据的采样和传输,因此SDIN数据信号必须在SCLK时钟的上升沿保持稳定。当处理器向显示驱动传输的是命令时需要将D/C引脚拉低,当传输的是数据时需要将其拉高处理。





当实现底层的显示驱动控制逻辑后,即可实现处理器对显示驱动的配置和数据的传输。在上电后需要按照显示驱动芯片手册中的要求执行如下操作,即处理器需要完成对应的配置后才可以传输数据从而显示。这部分可以参考芯片厂商提供的驱动,无需自行实现,只需根据需求修改配置即可。

上电配置和相关命令

### 在完成上电配置后即可向显示驱动发送数据,从而让OLED显示字符等。

Start-up sequence:

- 1. Power up VDD by pulling OLED\_VDD low. Wait 1ms.
- 2. Pulse RES# low for at least 3us.
- 3. Send initialization/configuration commands (see Table 12).
- 4. Power up VBAT by pulling OLED\_VBAT low. Wait 100ms for voltage to stabilize.
- 5. Clear screen by writing zero to the display buffer.
- 6. Send "Display On" command (0xAF).

Command function	Command bytes
Charge pump enable	0x8D, 0x14
Set pre-charge period	0xD9, 0xF1
Contrast control	0x81, 0x0F
Column inversion disable	0xA0
Scan direction	0xC0
COM pins configuration	0xDA, 0x00
Addressing mode: horizontal	0x20

Table 12. OLED configuration commands.



③板卡适配: GPIO按键中断

此处将PAD\_GPIO\_8分配至中间的按键,默认为低电平,当按键按下时该引脚为高电平。因此将该GPIO中断配置为上升沿触发,在中断处理程序中通过串口输出中断信息。

## Buttons

static void gpio\_interrupt\_handler(int32\_t idx)

printf("gpio interrupt is triggled\n");

void gpio\_falling\_edge\_interrupt(pin\_name\_e gpio\_pin)

软件设置为输入模式、上升沿触发中断

```
example_pin_gpio_init();
```

printf("press the center button will triggle the gpio interrupt \n"); pin = csi\_gpio\_pin\_initialize(gpio\_pin, gpio\_interrupt\_handler);

csi\_gpio\_pin\_config\_mode(pin, GPIO\_MODE\_PULLNONE); csi\_gpio\_pin\_config\_direction(pin, GPIO\_DIRECTION\_INPUT); csi\_gpio\_pin\_set\_irq(pin, GPIO\_IRQ\_MODE\_FALLING\_EDGE, 1);





set\_property -dict { PACKAGE\_PIN B22 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_GPIO\_8 }]; #IO\_L20N\_T3\_16 Sch=btnc

硬件设计

# FPGA实现流程 ③板卡适配: SD卡读写控制

处理器对SD卡进行读写通信操作一般有两种通信接口可选,一种是SPI接口,另外一种是SDIO 接口。此处采用GPIO软件模拟SPI协议读写SD卡。

下图是Nexys Video板卡上SD卡槽接口的原理图和SD卡的两种通信方式下的引脚定义。当 SD\_RESET引脚拉低时SD卡停止供电实现复位。CARD\_DETECT可用于检测SD卡是否插入卡 槽, 默认为高电平, 当SD卡插入卡槽时该引脚会在机械结构作用下与GND相连拉低。



#### ## SD card

set property -dict { PACKAGE PIN W19 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 6 }]; #IO L12P T1 MRCC 14 Sch=sd cclk PAD GPIO 6 sclk set property -dict { PACKAGE PIN T18 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 18 }]; #IO L20N T3 A07 D23 14 Sch=sd cd card detect set property -dict { PACKAGE PIN W20 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 5 }]; #IO L12N T1 MRCC 14 Sch=sd cmd mosi PAD GPIO 5 set property -dict { PACKAGE PIN V19 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_GPIO\_7 }]; #IO\_L14N\_T2\_SRCC\_14 Sch=sd\_d[0] PAD GPIO 7 miso #set property -dict { PACKAGE PIN T21 IOSTANDARD LVCMOS33 } [get ports { sd d[1] }]; #IO L4P T0 D04 14 Sch=sd d[1] #set property -dict { PACKAGE PIN T20 IOSTANDARD LVCMOS33 } [get ports { sd d[2] }]; #IO L6N T0 D08 VREF 14 Sch=sd d[2] set property -dict { PACKAGE PIN U18 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 19 }]; #IO L18N T2 A11 D27 14 Sch=sd d[3] PAD GPIO 19 CS set property -dict { PACKAGE PIN V20 IOSTANDARD LVCMOS33 } [get ports { PAD GPIO 20 }]; #IO L11N T1 SRCC 14 Sch=sd reset PAD GPIO 20 rst

硬件设计



在实际应用中直接操作SD卡存储单元是不现实的。SD卡一般用来存放文件,所以都需要加载文件系统到里面,此处移植FatFs文件系统进行对SD卡的控制。



这些应用函数的实现均在FatFs模块中,这部分只 需在ffconf.h文件中完成配置以满足不同应用即 可。需要用户实现的就是绿色部分的底层存储设 备的读写操作函数、存储设备信息获取函数等。



冬



## ④FLASH固化

Nexys Video板卡上搭载的 FLASH型号为S25FL256S, 通过上图的勾选生成bin文件 选项后即可在综合实现后生 成。由于FLASH读写速度相对 较慢,因此将bin文件写入 FLASH需要较长时间。

完成以上步骤后即可综合实 现生成bitstream文件并加载 入FPGA,但在FPGA掉电后 需要重新加载。因此可以通 过生成bin文件并加载入 FLASH存储,在FPGA上电 后自动从FLASH中读取并完 成配置。 🏊 rseponder - [D:/FPGA/rseponder/rseponder.xpr] - Vivad 💦 Tools Reports Window ø Σ Flow Navigator ₹ ≑ ? **PROJECT MANA** PROJECT MANAGER Sources Settings Q I Add Sources Design Language Templates > .... P Catalog Do Connte Hierarchy ✓ IP INTEGRATOR Properties Create Block Design Open Block Design Generate Block Design SIMULATION Tci Console Run Simulation QI ✓ RTL ANALYSIS Name synth 1 > Open Elaborated Design 🤞 impl √ synth\_2 ✓ SYNTHESIS ✓ impl\_: Run Synthesis > Open Synthesized Design

Settings					
Q- Project Settings General	Bitstream Specify various settings related to writing bitstream				
Simulation	(i) Note: Additional bitstream settings will be availa	ble once you open an imp			
Elaboration	✓Write Bitstream (write_bitstream)				
Implementation	tcl.pre				
Bitstream	tcl.post				
> IP	-raw_bitfile				
	-mask_file				
Tool Settings	-no_binary_bitfile				
Project IR Defaulte	-bin_file*	$\checkmark$			
P Delaulis Source Eile	-readback_file				
Display	-logic_location_file				
WebTalk	-verbose				
Help	More Options				
<ul> <li>&gt; Text Editor</li> <li>3rd Party Simulators</li> <li>&gt; Colors</li> <li>Solution Bulloc</li> </ul>					
Selection Rules Shortcuts > Strategies > Window Behavior	Select an option above to see a description of it				
?)	ок	Cancel Appl			

Restore...

implemented design



创建工程

## 在Vivado中创建一个新的工程

project\_1 - [D:/wujian100-test/project\_1/project\_1.xpr] - Vivado 2018.3 Tools Reports File Edit Flow Window Layout View

X

Com

Project		<u>N</u> ew	R
Add So <u>u</u> rces	Alt+A	Open	ict_1
<u>C</u> lose Project		Open <u>R</u> ecent	-
<u>C</u> onstraints	2	Open Ex <u>a</u> mple	
Simulation Waveform		Save As	<u>[3]</u>
Chec <u>k</u> point	×	Open Log File	
Īb			-
Text E <u>d</u> itor	*	Archive	
I <u>m</u> port		_	
Export	×		
Launch SDK			
<u>P</u> rint	Ctrl+P		
Exit		Hierarchy Libraries	Co

#### 🔥 New Project

#### Project Type

Specify the type of project to create.

#### <u>R</u>TL Project

You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis. ×

Do not specify sources at this time

<u>Post-synthesis Project</u>: You will be able to add sources, view device resources, run design analysis, planning and implementation.

Do not specify sources at this time

#### J/O Planning Project

Do not specify design sources. You will be able to view part/package resources.

) Imported Project

Create a Vivado project from a Synplify, XST or ISE Project File.

Example Project

Create a new Vivado project from a predefined template.

## (1) 注意要选择RTL Project

🔥 New Project

#### Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

$+$ $ -  \pm   \pm$							
	Index	Name	Library	HDL Source For		Location	
-	1	soc_file	xil_defaultlib	Synthesis & Simulation		D:/学习/RISC-V workshop/note	
-	2	head_file	xil_defaultlib	Synthesis & Simulation		D:/学习/RISC-V workshop/note	

<					-	;
		Add Files Ag	d Directories	<u>C</u> reate File		
Scan and add F	RTL <u>i</u> nclude files	into project				
Copy <u>s</u> ources in	nto project					
Add sources fro	om subdirectorie:	5				
Target language:	Verilog 🗸 🗸	Simulator language:	Mixed 🗸			

# (2) 添加设计源文件和头文件

4

 $\times$ 

### 🝌 New Project

1

### Add Constraints (optional)

Specify or create constraint files for physical and timing constraints.

+, - +	Ŧ
Constraint File	Location
NexysVideo.xdc	D:(学习\RISC-V workshop\note\wujian100_vivado\wujian_src\xdc

# (3) 添加约束文件

X

🔥 New Project

### Default Part

1

Choose a default Xilinx part or board for your project.

Parts	Boards							
Reset A	II Filters							
endor:	All	~	Name:	AII				
Search:	Q- Nexy			🔇 🗸 (1	match)			
Display	/ Name				Preview	Vendor	File Version	Part
Nexys	Video					digilentinc.com	1.1	xc7a200

# (4) 选择正确的板子

.....

×



1

修正错误、加载IP

(1) 有错误, 需修改类型

Sources ? _ 🗆 🗠 ×	Project Summary	
Q. 素 ≑ + 2 98 ✿	Overview   Dashboar	d
V Design Sources (9)		1
✓	Settings Edit	
apb0_params.y	Project name:	project_2
apb1_params.v	Project location:	D:/wujian100-test/proje
timers_params.v	Product family:	Virtex-7
wdt_params.v	Project part:	xc7vx485tffg1157-1
> 🔁 Non-module Files (4)	Top module name:	wujian100_open_top
> • • * wujian100_open_top (wujian100_open_fpga_top.v) (6	Target language:	Veriloo
Constraints (1)	Set Type	×
Hierarchy Libraries Compile Order	Set the type of the selected :	sources.
Source File Properties ? _ 🗆 🖸 🗙		
● apb0_params.v ← → 🌣	File type: Verilog Heade	ir 🗸
Enabled	ОК	Cancel
Location: D:/wujian100-test/project_2/project_2.srcs/sour	Report Strategy:	/ivado Synthesis Default R
Type: Verilog		
Library: xil_defaultlib ···		
Size: 3.5 KB	DRC Violations	
General Properties		Run Implementation

PROJECT MANAGER	Sources ? _ 🗆 🗹 🗙	Project Summary × wujian100_open_fpga_top.v × IP Catalog ×		
Settings	Q, , ¥,   ♦   +   ₪   ● 0 🔅	Cores   Interfaces		
Add Sources Language Templates	Design Sources (5)     Verilon Header (4)			
후 IP Catalog	v • * wujian100_open_top (wujian100_open_fpga_top.v) (6)	Search: Q- clk (3 matches)		
	u_clk_wiz_0:xil_defaultlib.clk_wiz_0	Name ^1 AXI4 Status License VLNV		
IP INTEGRATOR	> 🔵 x_aou_top : aou_top (aou_top.v) (3)	🗸 🗁 Vivado Repository		
Create Block Design	> 🔵 x_pdu_top : pdu_top (pdu_top.v) (4)	✓  ☐ Debug & Verification		
	>  x_cpu_top : core_top (core_top.v) (2)	Clock Verification IP     Prod Included xilin		
Open Block Design	> 🔵 x_retu_top : retu_top (retu_top.v) (1)	Simulation Clock Generator     Prod     Included xilin		
Generate Block Design	x_PAD_EHS : PAD_OSC_IO (PAD_OSC_IO.v)			
	Vierarchy Librarias Compile Order	Clocking		
SIMULATION		Clocking Wizard AXI4 Prod Included xilin		
RTL ANALYSIS	IP Properties ? _ $\Box$ $\boxtimes$ X			
	🌻 Clocking Wizard 🛛 🔶 🔅	9		
SYNTHESIS				
Run Synthesis	version. 0.0 (Rev. 2)			
> Onen Synthesized Design	Interfaces: AXI4	Details		
<ul> <li>Open synthesized besign</li> </ul>	Description: The Clocking Wizard creates an HDL file (Verilog	Name: Clocking Wizard		
	customized to the user's clocking requirements.	Version: 6.0 (Rev. 2)		
	Status: Production	Interfaces: AXI4		
Run Implementation	License: Included	Description: The Clocking Wizard creates an HDL file (Verilog or VHDL) that contains a clocking circuit customized to t		
> Open Implemented Design		Status: Production		

#### Component Name clk\_wiz\_0

Clocking Options	Output Clocks	s Port Renaming	MMCM Settings	Summary				
output civen	1 or chunne	Requested	Actual	Requested	Actual	Requested	Actual	DINGS
Clk_out1	clk_out1	20.000 🛞	20.000	0.000 🛞	0.000	<mark>50.000</mark>	50.0	BUFG
clk_out2	clk_out2	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
clk_out3	clk_out3	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
cik_out4	clk_out4	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
clk_out5	clk_out5	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
clk_out6	clk_out6	100.000	N/A	0.000	N/A	50.000	N/A	BUFG
Cik_out7	clk_out7	100.000	N/A	0.000	N/A	50.000	N/A	BUFG

#### USE CLOCK SEQUENCING

1

Output Clock	Sequence Number
clk_out1	1
clk_out2	1
clk_out3	1
clk_out4	1
clk_out5	1
clk_out6	1
clk_out7	1

Clocking	Feedba	ck
----------	--------	----

Source	Signaling	
Automatic Control On-Chip	Single-ended	
O Automatic Control Off-Chip	<ul> <li>Differential</li> </ul>	
O User-Controlled On-Chip		
O User-Controlled Off-Chip		

#### Enable Optional Inputs / Outputs for MMCM/PLL

reset power\_down input\_clk\_stopped

Active High 
 Active Low

Reset Type

locked clkfbstopped

sources	? _ C X Project Summary × wujian100_open_fpga_top.v ×	
Q 🗶 🜲 💠 🕂 😰 💿 0	D:/wujian100-test/project_2/project_2.srcs/sources_1/imports/wujian_src/soc_file/wujian100_open	_fpga_t
Design Sources (5) Seilog Header (4)		
v e wujian100_open_top (wujian100)		
u_clk_wiz_0 : xil_defaultlib.clk_v	NIZ_0 102 POUT_EHS,	
> v_aou_top : aou_top (aou_top.v	)(3) 103 vadj_en,	
> _ x_pdu_top : pdu_top (pdu_top.v	)(4) 105 ):	
> o x_cpu_top : core_top (core_top:	v) (2) 106	
> x_retu_top : retu_top (retu_top.v	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	
X_PAD_EHS : PAD_OSC_IO (PA	109 (	
Hierarchy Libraries Compile Order	110 // Clock out ports 20MHz	
	111 .clk_outl(FIM_BAS	
Source File Properties	? _ 🗆 🖾 × 113 .resetn(PAD_MCURST),	
wujian100_open_fpga_top.v	← → ↔ 114 // Clock in ports 100MHz	
Enabled	117	
Location: D:/wujian100-test/project_	2/project_2.srcs/sour 118 output reg vadj_en:	
Type: Verilog ····	120 ; <b>reg</b> [31:0] counter;	
	121 reg flag;	
Liprary: XII defaulting	192	
	122 ' nermeter time1 = 32' 400000. //the bigger and	
Size: 69.6 KB	123         parameter time1 = 32' d400000;         //the bigger one           124         parameter time2 = 32' d200000;         //the bigger one	

(2) 加时钟分频IP

1

Project Summary X wujian100\_open\_fpga\_top.v X IP Catalog X NexysVideo.xdc

D:/学习/RISC-V workshop/note/wujian100\_vivado/wujian\_src/xdc/NexysVideo.xdc

### Q, 🔛 ← → 🔏 🖬 🖬 🗙 🖊 🎟 🌻

1 ### This file is a general .xdc for the Nexys Video Rev. A

2 ### To use it in a project:

3 ; ### - uncomment the lines corresponding to used pins

#### - rename the used ports (in each line, after get\_ports) according to the top level signal names in the project

6 ¦

9 10

12

## Clock Signal

set\_property -dict { PACKAGE\_PIN R4 IOSTANDARD LVCMOS33 } [get\_ports { clk }]; #IO\_L13P\_T2\_MRCC\_34 Sch=sysclk
create\_clock -add -name sys\_clk\_pin -period 10.00 -waveform {0 5} [get\_ports clk]

11 set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets PAD\_JTAG\_TCLK]

13 ¦ ## FMC Transceiver clocks (Must be set to value provided by Mezzanine card, currently set to 156.25 MHz)

14 ; ## Note: This clock is attached to a MGTREFCLK pin

15 | #set\_property -dict { PACKAGE\_PIN E6 } [get\_ports { GTP\_CLK\_N }];

16 #set\_property -dict { PACKAGE\_PIN F6 } [get\_ports { GIP\_CLK\_P }];

17 ; #create\_clock -add -name gtpclk0\_pin -period 6.400 -waveform {0 3.200} [get\_ports {GTP\_CLK\_P}];

18 | #set\_property -dict { PACKAGE\_PIN E10 } [get\_ports { FMC\_MGT\_CLK\_N }];

19 #set\_property -dict { PACKAGE\_PIN F10 } [get\_ports { FMC\_MGT\_CLK\_P }];

20 | #create\_clock -add -name mgtclk1\_pin -period 6.400 -waveform {0 3.200} [get\_ports {FMC\_MGT\_CLK\_P}];

(3) 确认约束内容

板上端口

## 设计模块中的端口

set\_property -dict { PACKAGE\_PIN AA19 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_USI0\_SD0 }]; #I0\_L15P\_T2\_DQS\_RDWR\_B\_14 Sch=uart\_rx\_out
set\_property -dict { PACKAGE\_PIN V18 IOSTANDARD LVCMOS33 } [get\_ports { PAD\_USI0\_SCLK }]; #I0\_L14P\_T2\_SRCC\_14 Sch=uart\_tx\_in

#### ## Buttons

## UART

set_property	-dict	{ PACKAGE_PIN B2:	2 IOSTANDARD	LVCMOS33 }	[get_ports { PAD_GPI0_8 }]; #I0_L20W_T3_16 Sch=btnc
set_property	-dict	{ PACKAGE_PIN D2:	2 IOSTANDARD	LVCMOS33 }	[get_ports { PAD_GPI0_9 }]; #I0_L22N_T3_16 Sch=btnd
set_property	-dict	{ PACKAGE_PIN C2:	2 IOSTANDARD	LVCMOS33 }	[get_ports { PAD_GPI0_10 }]; #I0_L20P_T3_16 Sch=btnl
set_property	-dict	{ PACKAGE_PIN D14	IOSTANDARD	LVCMOS33 }	[get_ports { PAD_GPI0_11 }]; #I0_L6P_T0_16 Sch=btnr
set_property	-dict	{ PACKAGE_PIN F1	5 IOSTANDARD	LVCMOS33 }	[get_ports { PAD_GPI0_12 }]; #I0_0_16 Sch=btnu
set_property	-dict	{ PACKAGE_PIN G4	IOSTANDARD	LVCMOS15 }	[get_ports { PAD_MCURST }]; #I0_L12N_T1_MRCC_35 Sch=cpu_resetn

# FLASH固化流程





Device: 🍈 xc7a200t_0						
er		-				
Manufacturer	`		Туре	All		~
Density ( <u>M</u> b) All			Width	All		~
lect Configuration Memory Part	Ø	) (2 maiches	)			
Name	Part	Manufact	Alias	Family	Туре	Density
\$25fl256sxxxxx0-spi-x1_x2_x4	s25fl256sxxxxx0	Spansion		s25flxxxs	spi	256
s25fl256sxxxxx1-spi-x1_x2_x4	s25ii256sxxxxxx1	Spansion		s25flxxxs	spi	256
		-				
				-		;
(?)				0	к	Cancel
$\sim$						8











## I/O LAB实验主要完成操作:

- 串口输出调试信息
- LED1-LED3每秒闪烁一次
- OLED显示字符串
- ▶ 按下center button之后, 会触发GPIO中断从而输出信息。
- MCU在SD卡中新建文件并写入内容



NexysVideo\_Master.xdc - 记事本
 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

#### ## Pmod header JB

第

IOSTANDARD LVCMOS33 } [get ports { PAD PWM CH3 }]; #IO L21P T3 DQS 34 Sch=jb p[1] set property -dict { PACKAGE PIN V9 IOSTANDARD LVCMOS33 } [get ports { PAD PWM CH4 }]; #IO L21N T3 DQS 34 Sch=jb n[1] set property -dict { PACKAGE PIN V8 IOSTANDARD LVCMOS33 } [get ports { PAD PWM CH5 }]; #IO L19P T3 34 Sch=jb p[2] set property -dict { PACKAGE PIN V7 set property -dict { PACKAGE PIN W7 IOSTANDARD LVCMOS33 } [get ports { PAD PWM CH6 }]; #IO L19N T3 VREF 34 Sch=jb n[2] IOSTANDARD LVCMOS33 } [get ports { PAD PWM CH7 }]; #IO L24P T3 34 Sch=jb p[3] set property -dict { PACKAGE PIN W9 set property -dict { PACKAGE PIN Y9 IOSTANDARD LVCMOS33 } [get ports { PAD PWM CH8 }]; #IO L24N T3 34 Sch=jb n[3] set property -dict { PACKAGE PIN Y8 IOSTANDARD LVCMOS33 [get ports { PAD JTAG TMS }]; #IO L23P T3 34 Sch=jb p[4] IOSTANDARD LVCMOS33 ] [get ports { PAD JTAG TCLK }]; #IO L23N T3 34 Sch=jb n[4] set property -dict { PACKAGE PIN Y7



(1) 查看CKLINK与板子的连接方式



🕜 平头哥 status CKLink Lite TDI = • GND TDO • • GND TCK • • GND -- . . ..... nRST • • TMS -- • • --20 VREF • • TRST 1.2V≤Ext.VREF≤3.6V USB Power:5V/40mA Default Int. VREF=3.3V



a 2 🔲 📾 🖉 💽 🥥 🕫





# 打开project、下载程序

📕   🛃 📮   (	CDK			
文件 主页	共享 查看			
$\leftarrow \ \rightarrow \ \checkmark \ \uparrow$	<pre>« examples &gt; hello_world &gt; CDK </pre>	ひ /> 搜索"CDK"		
📌 快速访问	名称	修改日期	类型	大小
	.cdk	2021/6/8 16:30	文件夹	
OneDrive	📕 Lst	2021/6/8 15:24	文件夹	
华为云盘	📕 Obj	2021/6/15 10:28	文件夹	
	Makefile	2021/6/15 10:28	文件	
🍤 此电脑	wujian100_open-hello_world.cdkproj	2021/6/15 10:39	CDKPROJ文件	
📕 华为云盘	wujian100_open-hello_world.cdkws	2021/6/15 10:15	CDKWS 文件	
🍑 坚果云	wujian100_open-hello_world.mk	2021/6/15 10:28	<b>MK</b> 文件	
🧊 3D 对象	wujian100_open-hello_world.txt	2021/6/15 10:28	文本文档	

# (1) 用CDK打开project

4

🚾 [wujian100 open-hello world] D:\Amy document\documents\workshop\wujian100 sim\wujian100 open\sdk\projects\examples\hello world\main.c - [git: master] File Edit View SDK Project Flash Debug Peripherals Tools Windows Help 自ち・さ・くと「「馬馬馬 🖹 📄 🖸 📄 🗙 🔏 🖬 🛛 📝 🔛 🖸 📆 🍳 🔴 🖉 🖉 🕷 Project View [master] <sup>D</sup>X <global> v main(void) ✓ ① ぷ ③ an main.c X wujian100 ope ~ BuildSet \* Copyright (C) 2017-2019 Alibaba Group Holding Limited 🖵 wujian100 open-hello world \*/ wujian100 open-hello worl > 📒 board 6 > csi core \* @file main. c v 📄 csi driver \* @brief hello world \* @version V1.0 > 📒 include 10 \* @date 17. Jan 2018 ✓ i wujian100 open 11 > include 12 13 c devices.c #include <stdio.h> 14 #include"key.h" isr.c 15 #include "led.h C lib.c 16 #include "oled128\_32.h" C novic irg tbl.c 17 #include "ff.h" /\* Declarations of FatFs API \*/ 18 c pinmux.c 19 20 21 /\* FatFs work area needed for each volume \*/ FATFS FatFs; startup.S /\* File object needed for each open file \*/ FIL Fil; sys\_freq.c 22 23 extern void mdelay(uint32\_t ms); System.c void sd\_test(); c trap c.c 24 25 26 27 28 29 30 31 32 33 34 35 36 37 int main(void) vectors.S wj dmac v2.c printf("Hello World!\n"); wj irq.c key\_gpio\_intr(PA8); wj\_oip\_gpio.c LED\_Init() OLED\_SHOW (); wj\_oip\_timer.c sd\_test(); wj\_oip\_wdt.c while(1) wj\_pwm.c LED ON () wj rtc.c mdelay(500) C wj usi.c LED\_OFF () mdelay(500) wj\_usi\_iic.c 38 39 40 wj usi spi.c return 0; wj\_usi\_usart.c 41 42 43 44 45 46 47 48 49 wj\_usi\_wrap.c void sd\_test() > libs ✓ projects UINT bw: FRESULT fr; examples ✓ hello world > configs f\_mount(&FatFs, "", 0); /\* Give a work area to the default drive \*/ > key\_gpio\_intr 50 51 52 fr = f\_open(&Fil, "newfile.txt", FA\_WRITE | FA\_CREATE ALWAYS); /\* Create a file \*/ V ED if (fr == FR\_OK) C led.c f\_write(&Fil, "It works!\r\n", 11, &bw); /\* Write data to the file \*/ 53 fr = f\_close(&Fil); /\* Close the file \*/ h led.h 54 55 56 57 if (fr == FR\_OK && bw == 11) { > i oled128 32 printf("sd card success\r\n"); > sd driver C main.c 58

## (2) 在CDK中打开main.c

16

(3) 打开串口界面,设置串口

1

Proj

> wujia

er 🗸	Show Status Bar Show ToolBar		
2	Toggle Current Fold	Alt-RIGHT	
	Toggle All Folds	Ctrl-Alt-RIGHT	
1	Toggle All topmost Folds in Selection		7-2019 Alibaba Group Holding Limited
ji	Toggle Every Fold in Selection		
Ł	Display EOL		
c	Show Whitespace	>	
c li	Full Screen	Alt-M	orld
F	Show Welcome Page		2018
~	Load Welcome Page at Startup		
~	Output Pane	Ctrl-`	
~	Project Pane	Ctrl-Alt-W	b"
~	Navigation Bar	Ctrl-Alt-N	/* Declarations of FatFs API */
	Debugger Pane	Ctrl-Alt-D	(* FatFe work area needed for each volume */
	Frame Pane	Ctrl-1	/* File object needed for each open file */
	Peripherals Pane	Ctri-2	(a+22 + mg);
~	Serial Pane	Ctrl-3	L1032_0 m87,
	OSTrace Pane	Ctrl-4	
	Analysis Pane	Ctrl-5	
	Toolbars	>	rld!\n"); B);
	Toggle All Panes	Ctrl-M	

🚾 [ wujian100 open-hello world ] D:\Amy document\documents\workshop\wujian100 sim\wujian100 open\sdk\projects\examples\hello world\ma File





点击一次开始调试,再点击一次终止调试,此时程序就下进板子里了





Serial Pane Connected. Hello World! press the center button will triggle the gpio interrupt OLED\_SPI\_INIT sd card success

# (1) 首先观察串口界面的调试信息

## (2) 观察LED闪烁情况和OLED显示内容

## while(1)

LED\_ON(); mdelay(500); LED\_OFF(); mdelay(500);

### LED控制程序

void OLED\_SHOW()

```
OLED_Init();//初始化OLED
OLED_ColorTurn(0);//0正常显示,1 反色显示
OLED_DisplayTurn(1);//0正常显示 1 屏幕翻转显示
OLED_Clear();
OLED_ShowString(0,0,"Welcome to use",16);
OLED_ShowString(16,2,"WujianSoC",16);
```

OLED显示程序

### Serial Pane

### Connected.

Hello World!

press the center button will triggle the gpio interrupt OLED\_SPI\_INIT

sd card success

gpio interrupt is triggled

# (3) 按center button测试中断功能

# (4) 板子断电拔出SD卡,在电脑上查看是否写入内容

