Department of Computer Science & Engineering



# 计算机系统结构实验指导书-LAB2

### **1. OVERVIEW**

1.1 实验名称

## FPGA 基础实验: 4-bit Adder

1.2 实验目的

- 1. 掌握 Xilinx 逻辑设计工具 Vivado 的基本操作
- 2. 掌握 VerilogHDL 进行简单的逻辑设计
- 3. 使用功能仿真
- 4. 约束文件的使用和直接写法
- 5. 生成 Bitstream 文件
- 6. 上板验证(暂不做)
- **1.3** 实验预计时间 120分钟
- 1.4 实验报告与验收办法

本实验报告和工程文件以及之后的都在第十一周星期二晚上 23 点前提交

PS:为方便使用,可建立 C:\Archlabs 文件夹,用于放置实验工程和给定的 IP 核等

### **2. EXPERIMENTAL STEPS**

#### 2.1 新建工程

- 1. 启动桌面 Vivado 2018.3 开发工具
- 2. 点击 Create Project
- 3. 弹出 New Project, 建立一个新工程, 点击 Next

4. 输入工程名称 lab02, 工程位置建议 E:/archlabs, 确认勾选中 Create project subdirectory 后点击 Next

PS: 工程名称和存储路径中不能出现中文和空格,建议工程名称以字母、 数字、下划线来组成

new Project	×
Project Name Enter a name for your project and specify a directory where the project data files will be stored.  Project name: Iab02 Project jocation: E/archlabs Create project subdirectory Project will be created at: E/archlabs/lab02	
(?) <back next=""></back>	inish Cancel

5. 选择 RTL Project 工程类型, 勾选 Do not specify sources at this time, 点击 Next

A New Project	×
Project Type Specify the type of project to create.	4
RTL 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.	
₽ o not specify sources at this time	
Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.	
Do not specify sources at this time	
I/O Planning Project Do not specify design sources. You will be able to view part/package resources.	
<ul> <li>Imported Project</li> <li>Create a Vivado project from a Synplify, XST or ISE Project File.</li> </ul>	
Example Project Create a new Vivado project from a predefined template.	
Image: Second	Cancel

图 2-1 RTL 工程

6. 选择实验板的 FPGA 参数: Family 选 Kintex-7, Package 选 ffg676, Speed grade 选-2, 在具体型号中选 xc7k325tffg676-2, 点击 Next

Reset All	Filters								
Category:	All		~	Package: ffg	676	✓ Temper	ature: All Rei	maining	~
arriny.	Nintex-7		~	opeed2		¥			
Search:	Q-		~	×					
Part		I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	Ultra RAMs	DSPs	G
xc7k160	ffg676-2	676	400	101400	202800	325	0	600	8
xc7k325	ffg676-2	676	400	203800	407600	445	0	840	8
	100/0-2	0/0	400	254200	508400	(90	U	1540	8

7. 点击 Finish 结束工程的创建

-		×
VIVADO HLI Editori	New Project Summary  A new RTL project named Tab01' will be created.  The default part and product family for the new project. Default Part xc?X32580576-2 Partity: Kintes-7 Partity: Kintes-7 Package: ftg676 Speed Grade:-2	
E XILINX.	To create the project, click Finish  Kgack  Next Press	Cancel

#### 2.2 添加文件

1. 如下图,点击 左侧区 Flow Navigator 下的 Project Manager->Add Sources 或中间区 Sources 的 "+"号,打开 Add Sources 对话框



 选择第二项 Add or Create Design Sources,用来添加或新建 Verilog HDL 源文件,点击 Next

- 3. 若已有源文件或内核文件,可选 Add Files 项以添加文件。这里是要新建 1 位全加器模块文件,选择 Create File
- 4. 弹框 Create Source File 中输入 adder\_1bit 文件名,点击 OK

À Add Sources		×
Add or Create E Specify HDL, netlist disk and add it to yo	Design Sources , Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on our project.	2
+ +	A Create Source File X	
	Create a new source file and add it to your project.	
	Eile type: Verilog	
	File location: Society Construction:	
	OK         Cancel         s         Create File	
Scan and add	RTL include files into project	
Copy sources	into project	
Add so <u>u</u> rces f	from subdirectories	
?	< <u>B</u> ack <u>N</u> ext > Einish Cance	ł

- 5. 点击 Finish
- 6. 在弹出的 Define Module 中输入设计模块所需的端口,并设置端口方向; 完成后点击 OK

A 0	À Define Module 🛛 🗙								
Define a module and specify I/O Ports to add to your source file. For each port specified: MSB and LSB values will be ignored unless its Bus column is checked. Ports with blank names will not be written.									
	Module name	· adder 1b	it						
	Module name	auder_10	n					0	
	I/O Port Defini	itions							
	+  -	<b>↑</b>   ↓							
	Port Name	Direction	Bus	MSB	LSB	_			
	а	input 🗸		0	0			^	
	b	input 🗸		0	0				
	ci	input 🗸 🗸		0	0				
	s	output 🗸		0	0				
	со	outp 🗸		0	0				
6	)	input					OK	Cancel	
Ċ		output						Gancer	
		inout							

7. 在源代码编辑区双击 adder\_1bit, 添加如下方框中的代码

```
2
     module adder lbit(
 3
        input a,
 4
         input b,
         input ci,
 5
  6
         output s,
 7
         output co
 8
        );
         wire s1, c1, c2, c3;
 9
 10
         and (cl, a, b),
              (c2, b, ci),
 11
12
              (c3, a, ci);
13
14
             (sl, a, b),
         xor
15
              (s, sl, ci);
16
17
         or
              (co, cl, c2, c3);
18
19 endmodule
20
```

8. 接下来要实现 4 位加法器。继续创建源文件,命名为 adder\_4bits,点击 OK

A Create Source File									
Create a new source file and add it to your project.									
<u>F</u> ile type:	Verilog	~							
F <u>i</u> le name:	adder_4bits	$\otimes$							
Fil <u>e</u> location:	😜 <local project="" to=""> 🛛 🗸 🗸</local>								
?	ОК	Cancel							

9. 点击 Finish,输入模块所需的端口,并设置端口方向,若端口属总线型,勾选 Bus,并由 MSB 和 LSB 确定宽度点击 OK

🝌 D	A Define Module ×									
Define a module and specify I/O Ports to add to your source file. For each port specified: MSB and LSB values will be ignored unless its Bus column is checked. Ports with blank names will not be written.										
Module Definition										
	Module name	: adder_4b	its				8			
	I/O Port Defini	tions								
	+  -	t 🗐								
	Port Name	Direction	Bus	MSB	LSB					
	а	input 🗸	<	3	0		^			
	b	input 🗸	✓	3	0					
	ci	input 🗸 🗸		0	0					
	s	output 🗸	$\checkmark$	3	0					
	со	output 🗸		τ ο	0		~			
?	)					ОК	Cancel			

10. 双击 adder\_4bits, 添加如下代码

input [3:0] a,
input [3:0] b,
input ci,
output [3: 0] s,
output co
);
wire [2:0] ct;
adder_lbit al(.a(a[0]), .b(b[0]), .ci(ci), .s(s[0]),.co(ct[0])),
a2(.a(a[1]), .b(b[1]), .ci(ct[0]), .s(s[1]),.co(ct[1])),
a3(.a(a[2]), .b(b[2]), .ci(ct[1]), .s(s[2]),.co(ct[2])),
a4(.a(a[3]), .b(b[3]), .ci(ct[2]), .s(s[3]),.co(co));

# 2.3 功能仿真

1. 创建激励测试文件。在中间区 Sources 栏点击 "+"号或于左 侧区 PROJEU MANAGER 下选择 Add Source

2. 在 Add Sources 选择 Add or Create Simulation Source, 点击 Next

À Add Sources		×
HLX Editions	Add Sources This guides you through the process of adding and creating sources for your project Add or greate constraints Add or create design sources Add or create gimulation sources	
<b>£</b> XILINX.		
?	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish	Cancel

- 3. 选择 Create File 创建一个仿真激励文件
- 4. 激励文件测试名可输入 adder\_4bits\_tb, 点击 OK

À Create Source File >									
Create a new so project.	4								
<u>F</u> ile type:	Verilog	~							
File name:	adder_4bits_tb	$\otimes$							
Fil <u>e</u> location:	😜 <local project="" to=""></local>	~							
?	ОК Са	ancel							

- 5. 点击 Finish, 创建激励文件不需要对外端口, 点击 OK
- 6. 在弹出的对话框点击 YES

7. 在 Source 区 Simulation Sources 下,打开测试文件 adder\_4bit\_tb,在其中对 要仿真的模块进行实例化和激励代码的编写(并点击保存),如下图所示

```
2
    module adder_4bits_tb( );
 3
 4
         reg [3:0] a;
 5
        reg [3:0] b;
 6
        reg ci;
 7
 8
        wire [3:0] s;
 9
        wire co:
10
11
        adder 4bits u0 (
12
            .a(a),
13
            .b(b),
            .ci(ci),
14
            .s(s),
15
16
             .co(co)
17
            );
18
19
        initial begin
            a = 0;
20
21
            b = 0;
            ci = 0;
22
23
24
            #100;
25
            a = 4'b0001;
            b = 4'b0010;
26
27
            #100;
28
            a = 4'b0010;
29
            b = 4'b0100;
30
31
            #100;
32
            a = 4'bllll;
33
            b = 4'b0001;
34
            #100;
35
            ci = 1'b1;
36
37
         end
38
39 endmodule
```

8. 点击左侧区的 Run Simulation 并选择 Run Behavioral Simulation。下图为仿 真运行后得到的仿真波形图样例:

adder_1bit.v	× adder_4bits	.v × adder_4bits	_tb.v × Untitled	1* ×					2013
Q, 💾 🦉	Q 23	•F   I4   >I   12	±r +F Fe	+					0
						400.000 ns			^
Name	Value	10 ns	100 ns	200 ns	300 ns	400 ns	500 ns	600 ns	1700 ns 18
> 😻 a[3:0]	1111	0000	0001	0010			1111		
> 😻 b[3:0]	0001	0000	0010	0100			0001		
🕌 ci	1								
> 🐨 s[3:0]	0001	( 0000 )	0011	0110	0000		01	001	
🔓 co	1								
					es.				

观察分析可知,仿真结果与逻辑功能一致,电路能正常工作。如果不一致 可以放大或缩小、可以滑动黄色的时间运行竖线来查看输入与输出之间的 变化是否吻合,若有误可再次修改逻辑代码再仿真。

PS: 此处仿真为本实验一个检查点, 可截图保存写入报告

#### 2.4 工程实现

在本实验中要用实验板上的 8 个 Switch 对应二组 4 位二进制输入,用 4 个 LED 发光二极管对应输出,并用 2 个七段数码管显示运行结果。 故本实验需要用到 display.v 这个七段数码管 SEGMENT 和 LED 发光二 极管显示模块(实验室提供 display 核,以网表文件形式给出)

1. 将4位加法器的输出赋予LED和七段数码管显示,需要创建一个顶层源文件,可命名Top,如下图

sk and add it to your p	♣ Create Source File		×	
$ \mathbf{+}_{\mathbf{z}}  =   \mathbf{+}  $	Create a new source file and ad project.	d it to your	·	
	<u>F</u> ile type: • Verilog	~		
	F <u>i</u> le name: Top	8		
	Fil <u>e</u> location: 🕞 <local pre<="" td="" to=""><td>oject&gt; 🗸</td><td>File buttons below</td><td></td></local>	oject> 🗸	File buttons below	
	(?) ОК	Cancel		
	<u>A</u> dd Files	A <u>d</u> d Directorie	s <u>C</u> reate File	
Scan and add RT	_ include files into project		►	
Copy <u>s</u> ources into	project			
Add sources from	subdirectories			

进入Define Module,这里我们略过端口定义,后可在源程序中自行添加。双击 Top模块,添加以下端口变量和代码语句:

module Top(

input clk\_p, input clk\_n, input [3:0] a, input [3:0] b, input reset, output led\_clk, output led\_do, output led\_en, output wire seg\_clk,

```
output wire seg_en,
   output wire seg do
);
     wire CLK_i;
     wire Clk_25M;
    IBUFGDS IBUFGDS_inst (
           .O(CLK i),
           .I(clk p),
           .IB(clk n)
     );
     wire [3:0] s;
     wire co;
     wire [4:0] sum;
     assign sum = {co, s};
     adder 4bits Ul (
     .a(a),
     .b(b),
     .ci(1'b0),
     .s(s),
     .co(co)
     );
     reg [1:0] clkdiv;
     always@(posedge CLK i)
        clkdiv<=clkdiv+1;
     assign Clk 25M=clkdiv[1];
   display DISPLAY (
    .clk(C1k_25M),
    .rst(1'b0),
    . en(8' b0000011),
    .data({27'b0, sum}),
    .dot(8'b0000000),
.led(~{11'b0, sum}),
    .led_clk(led_clk),
    .led_en(led_en),
    .led_do(led_do),
    .seg_clk(seg_clk),
    . seg_en(seg_en),
    . seg_do(seg_do)
ΙP
   );
```

```
Endmodule
```

2. Top.v中用到了一个display IP核(已转换为网表),接下来需要添加该 核。如下图,在Add Sources 中,点击Add Sources:

Add Sources	×
Add or Create Design Sources Specify HDL, netlist, Block Design, and IP files, or directories containing those file types to add to your project. Create a new source file on disk and add it to your project.	4
$ +_{\lambda}  =  \pm  \mp$	
Use Add Files, Add Directories or Create File buttons below	
Add Files Add Directories Create File	
Scan and add RTL include files into project	
Copy sources into project	
✓ Add sources from subdirectories	
< Back     Next >     Einish     Ca	ancel

在Add Source	Files中同时选择一	·对与该核同前缀名的端口和网表文件,	如下图:
-------------	-------------	--------------------	------

🝌 Add Sourc	e Files		×
Look <u>i</u> n:	src	★ 会 果 土 A 陸 X C	
🚺 display.ed	f	Recent Directories	
display.v		D:/archlabs/lab02	~
		File Preview	
		(edif display	^
		(edifversion 2 0 0)	
		(edifLevel 0)	
		(keywordmap (keywordlevel 0))	
		(status	
		(written	
		(timeStamp 2019 03 11 14 17 49)	
		(program "Vivado" (version "2015.4"))	
		(comment "Built on 'Wed Nov 18 09:43:45 MST 2015'")	
		(comment "Built by 'xbuild'")	
		(Library hdi_primitives	
		(edifLevel 0)	× *
L			*
File <u>n</u> ame:	"display.edif" "display.v"		
Files of type:	Design Source Files (.vhd, vhdl, vhf, vhdp, vho, v, vf, verilog, vr, vg, vb	tf, vlog, vp, vm, veo, svo, vh, h, svh, vhp, svhp, edn, edf, edif, ngc, sv, sv	р У
		ок	Cancel

点击Finish,则在Sources区的顶层源文件下关联了这一对源文件

3. 添加约束文件。打开Add Sources对话框,如图选择第一项

🝌 Add Sources	×
HLx Editions	Add Sources This guides you through the process of adding and creating sources for your project Add or create constraints Add or create design sources Add or create gimulation sources
<b>£</b> XILINX.	
?	< <u>Back</u> Einish Cancel

Add Sources	X
Add or Create Constraints Specify or create constraint files for physical and timing constraint to add to your projection	ect.
Specify constraint set: 🕞 constrs_1 (active)	▲ Create Constraints File
+_  =   *   *	Create a new constraints file and add it to your
Use Add Files or Create File bu	Eile type:       INDC         File name:       Iab02_xdc         File location:       <-Local to Project>
Add Files Copy constraints files into project	OK     Cancel       File     Image: Second
(?)	Back Next > Einish Cancel

如下图位置,打开这新建的空白的约束文件:



#### 本实验添加如下约束代码:

set\_property PACKAGE\_PIN AC18 [get\_ports clk\_p]
set\_property IOSTANDARD LVDS [get\_ports clk\_p]

set property PACKAGE PIN AA12 [get ports {a[3]}] set property PACKAGE PIN AA13 [get ports {a[2]}] set property PACKAGE PIN AB10 [get ports {a[1]}] set property PACKAGE PIN AA10 [get ports {a[0]}] set property IOSTANDARD LVCMOS15 [get ports {a[0]}] set property IOSTANDARD LVCMOS15 [get\_ports {a[1]}] set property IOSTANDARD LVCMOS15 [get ports {a[2]}] set property IOSTANDARD LVCMOS15 [get\_ports {a[3]}] set property PACKAGE PIN AD10 [get ports {b[3]}] set property PACKAGE PIN AD11 [get ports {b[2]}] set\_property PACKAGE\_PIN Y12 [get\_ports {b[1]}] set property PACKAGE PIN Y13 [get ports {b[0]}] set property IOSTANDARD LVCMOS15 [get ports {b[0]}] set property IOSTANDARD LVCMOS15 [get ports {b[1]}] set property IOSTANDARD LVCMOS15 [get ports {b[2]}] set\_property IOSTANDARD LVCMOS15 [get\_ports {b[3]}] set property PACKAGE PIN N26 [get ports led clk]

set\_property PACKAGE\_PIN M26 [get\_ports led\_do]
set\_property PACKAGE\_PIN P18 [get\_ports led\_en]
set\_property IOSTANDARD LVCMOS33 [get\_ports led\_clk]
set\_property IOSTANDARD LVCMOS33 [get\_ports led\_do]
set\_property IOSTANDARD LVCMOS33 [get\_ports led\_en]

set\_property PACKAGE\_PIN M24 [get\_ports seg\_clk]

```
set_property PACKAGE_PIN L24 [get_ports seg_do]
set_property PACKAGE_PIN R18 [get_ports seg_en]
set_property IOSTANDARD LVCMOS33 [get_ports seg_clk]
set_property IOSTANDARD LVCMOS33 [get_ports seg_do]
set_property IOSTANDARD LVCMOS33 [get_ports seg_en]
```

# 2.5 下载验证 (暂不做)

在 Flow Navigator区点击 Program and Debug下Generate Bitstream选项,系统将自动完成综合、实现、并生成 FPGA 配置文件。出现以下框,点Yes

PROGRAM AND DEBUG     Generate Bitstream	Type:         XDC         30         set_property PACKAGE_PIN L24           31         set property PACKAGE PIN R18           No Implementation Results Available	[get_p [get p X	101 101
> Open Harware Manage			t
	<ul> <li>There are no implementation results available. OK to launch synthesis and implementation 'Generate Bitstream' will automatically start when synthesis and implementation completes.</li> <li>Don't show this dialog again</li> </ul>	?	t

接下来的框, 点OK

🝌 Launch Runs	×
Launch the selected synthesis or implementation runs.	
Launch <u>d</u> irectory: 🕞 <default directory="" launch=""> 🗸 🗸</default>	
Options	
● Launch runs on local host: Number of jobs: 4 ~	
◯ <u>G</u> enerate scripts only	
Don't show this dialog again	
OK Cancel	

FPGA 配置文件生成后,便可将bit文件下载到实验板上以查看真实效果, 此次选择第三项直接去"烧写"

Bitstream Generation Completed	Х
Bitstream Generation successfully completed.	
Open Implemented Design	
○ <u>V</u> iew Reports	
Open <u>H</u> ardware Manager	
○ <u>Generate Memory Configuration File</u> 打开硬件程序和调试管理	]
Don't show this dialog again	
OK Cancel	]

- 2. 连接好实验板的电源和JTAG下载线,然后打开开关
- 3. 将bit 文件下载到实验板以查看真实效果。点击 Flow Navigator 中点击 Open Hardware Manager下的Open Target再选中Auto Connect



 点击Open Hardware Manager下的Program Device,再选择FPGA芯片 xc7k325t\_0,弹出的对话框中文件名处会自动加载本工程生成的bit文件,点击 Program对FPGA 芯片进行现场编程

🍌 Program Device		×
Select a bitstream prog select a debug probes f programming file.	ramming file and download it to your hardware device. You can optionally file that corresponds to the debug cores contained in the bitstream	4
Bitstre <u>a</u> m file:	D:/archlabs/lab02/lab02.runs/impl_1/Top.bit	•••
Debug probes file:	•	•••
✓ Enable end of st	artup check	
?	<u>P</u> rogram Can	icel

5. 观察实验结果,显示是否符合预期。等待bit文件下载配置完成,尝试 拨动开关。拨码开关的低4位和次高4位代表两个加数;LED的低4位代表 的是和,第5位代表的是进位;2个七段数码管也代表和的结果。