基于FPGA的cy7c68013a双向通信实验



基于FPGA的cy7c68013a双向通信实验

本实验是基于FPGA的cy7c68013a的USB双向通信实验,以前折腾过一段时间cy7c68013a,没有入门时感觉 好难,入门了就会感觉很简单。本教程主要内容:

1.cy7c68013a的固件编写,以及生成iic固件和下载固件。

2.cy7c68013a的slave模式,以及他的读写时序

3.cy7c68013a的FPGA的上板测试,包括发送和接受两部分

一、驱动

在进行试验前要先安装好cypress提供的usb驱动,插上usb后,电脑就会检测到未识别的设备,这时打开设备管理器,右键未识别的usb,然后手动选择驱动,驱动会在本教程最后的链接中给出



请选择您的搜	浏览文件夹 🥐 🔀	En
	选择包含您的硬件的驱动程序的文件夹。	S)
 在这些的 		
使用下列 到的最佳	⊞ ☐ CyLoad ▲ ☐ CyMonfx1_fx21p	会安装找
□搜	☐ cyusbfx1_fx21p ₩ C w2k	
☑在	🗄 🧰 wlh-vista	
A		P
 不要搜索 选择这个 	₩54 ₩86	选择的驱
动程序与	E CPIF Derimer	
	Hardware	
	要查看任何子文件夹,诸单击上面的 + 号。	取消
	确定取消	

二、固件

cy7c68013a想要正常工作,就得给他编写好相应的固件,然后再固化到其内部,当然PC也是需要安装相应的驱动的,固件的编写主要是确定IN和OUT端点,以及一些标志信号。



固件只需要改这些参数,这里我都写好了,大家就不需要再改了,很容易看出我设置的时钟是48MHz,,然后 设置

EP2为OUT端点,512字节,4缓冲,bulk (注意OUT,IN都是相对PC来说的,OUT表示PC--->cy7c68013a,IN则相反)

EP6为IN端点,512字节,4缓冲,bulk

flag_a为EP2的EF,也就是空标志信号,为低时表示空,也就是没有数据过来,为高则表示有数据来了

flag_d为EP6的FF,也就是满标志信号,为低时表示写满了,这时再去写就是无效写了,为高则表示没有写满,可以继续写。





https://blog.csdn.net/chengfengwenalan

我写好的固件所在文件夹:固件源码什么的都在Firmware文件中

· · · · ·

st_usb → doc	> Firmware > output			
名称	~	修改日期	类型	大小
🐻 dscr.obj		2018/5/10 21:55	OBJ 文件	2 KB
🐻 fw.obj		2018/5/10 21:55	OBJ 文件	56 KB
📧 Hex2bix.exe	e	2012/6/7 14:06	应用程序	40 KB
🐻 periph.obj		2018/5/10 21:55	OBJ 文件	59 KB
🗋 usb		2018/5/10 21:55	文件	120 KB
📀 usb.build_la	og.htm	2018/5/10 21:55	Chrome HTML D	2 KB
📄 usb.hex	类型: Chrome HTML Docum	ent 8/5/10 21:55	HEX 文件	7 KB
🗋 usb.iic	大小: 1.31 KB	8/5/10 21:55	IIC 文件	2 KB
📄 usb.lnp	1%成口朔:2018/5/1021:55	8/5/10 21:55	LNP 文件	1 KB

https://blog.csdn.net/chengfengwenalan

三、slave的写时序



Figure 9-12. Timing Example: Synchronous FIFO Writes, Waveform 1

有图很容易看出,再写之前要先把FIFOADR确认好,这个决定了你写的对象是谁

Table 9-2. FIFO Selection via FIFOADR[1:0]

FIFOADR[1:0]	Selected FIFO
00	EP2
01	EP4
10	EP6
tns•//h 1 hg_csdn	net/ch EP8 fengwa

然后在该fifo非满时(相应的FF标志位高),才可以进行写操作,这个时序很简单,就是拉低slwr信号就可以 了,注意FD要与slwr对齐。

注意: 写操作时, slwr与FD的数据都是FPGA来控制的,为了让cy7c68013a更好的采样, ifclk与clk反向之后再 发送给cy7c68013a.

asslignps:jfclkg.csdn.n∉t/chengfeckkenalan

四、slave读操作

Figure 9-17. Timing: Synchronous FIFO Reads, Waveform 1



读时序跟写也是类似的,再读之前先确定FIFOADR,然后拉低sloe,这时FD总线就会出现第一个数据,然后检测到slrd为低时,FD就会显示下一个数据。

五、FPGA与cy7c68013a通信

前面主要是一些准备工作,现在开始进入正式的通信过程,项目工程如下:

test_	usb⇒				
名	称	^	修改日期	类型	大小
	doc	驱动, API, 固件, datasl	1 20 \$8/5/11 15:32	文件夹	
	prj	FPGA项目工程目录	2018/5/11 15:33	文件夹	
	signal_tap	调试	2018/5/10 20:17	文件夹	
	sim	仿直	2018/5/9 21:23	文件夹	
	src	FPGA源码	2018/5/10 22:48	文件夹	

https://blog.csdn.net/chengfengwenalan



具体代码都已经有了详细注释了,自己去看看代码就知道了,这里就不详细说了

本教程所用的调试工具是官方的工具

o → doc → CyConsole					
	修改日期	类型	大小		
CyConsole.chm	2011/10/18 12:03	编译的 HTML 帮	280 KB		
yConsole.exe	2011/10/18 12:03	应用程序	953 KB		
yconsole.pdf	2011/10/18 12:03	Foxit Reader PD	1,354 KB		
₹ 7 Cypress U	SB Console		- 🗆 🗙		
File Options	Help				
	🗐 🕒 Selected Script:		X 🕀 🐨 🖗		
Select Device	Davias Nama	News in Mindows Day	ing May Group in O		
	Device Name	Name in windows Dev	ice Mgr (from .inr)		
😙 EZ-USB Interface				- 🗆	×
Device	Clear L	oad Mon SEEPROM	Select Mon		
Get Dev Get Conf Get Pipe	Get Strings Download	Re-Load Lg EEPROM	URB Stat HOLD	RUN	
Vend Reg Reg Ov00 Value 0		oth 0 Dir 0 OUI	Hex Bytes CO B	A 0A 81 00 0	1.00 -
Take here inter the terms of t				+ 0+ 01 00 0	
Iso Trans Pipe	Length 128	Packet Size	Packets		
Bulk Trans Pipe	▼ Length 64	Hex Bytes 5		•	
Reset Pipe Abort Pipe File Trans	Pipe	-			
Set IFace Interface 0 Alt	Setting 0				
					<u> </u>
					\sim
					et/chengt

下面给出写的signal tap 的调试截图

写是一次写512个字节数据,0-255,注意usb的fifo是一次发送16位的,也就是2个字节。先发送低字节,然后再 发送高字节,这我直接把低字节给赋值为0了

assign	usb_data	(state_c[2]	== 1'b1) ?	? {cnt,8	8'h00}	: 16'	hzzzait/6先发送低字苏,	n然后再发送高字节n
	_			· · ·				

trigge	er: 201	8/05/10 22:52:05 #0	Lock mode:	<mark>-</mark> Allow all chan	ges
		Node	Data Enable	Trigger Enable	Trigger Conditions
Туре	Alias	Name	61	61	1 🗹 Basic AND 🔹
*		flag_a	\checkmark	\checkmark	
*		flag_d	\checkmark	\checkmark	
*		sloe	\checkmark	\checkmark	
*		slrd	\checkmark	\checkmark	
*		slwr	\checkmark	✓	<u>0</u>
*		fifo_addr[10]	\checkmark	\checkmark	Xh
i		usb_data[150]	\checkmark	\checkmark	XXXXh
*		usb:usb_inst cmd_flag	\checkmark	\checkmark	
5		usb:usb_inst cmd_data[150]	\checkmark	\checkmark	XXXXh
-			\checkmark	\checkmark	XXh
		usb:usb_inst state_c[20]	\checkmark	\checkmark	Xh
		⊡_usb:usb_inst cnt0[9.10];ps://b]	og. 🗹 sdn.	net/⊠heng	f xxxxxxxxxbn

log: 1	Frig @ 3	2018/05/11 20:14:32 (0:0:1.3 elapsed)								
Туре	Alias	Name	-256	-128	. 9	128	256	384	512	
*		flag_a								
*		flag_d								
*		sloe								
*		slrd								
-		slwr								
*		fifo_addr[10]								
1				FFFFh						
-		usb:usb_inst cmd_flag								
6		⊡ usb:usb_inst cmd_data[150]		FFFFh						
-		⊡ usb:usb_inst cnt[70]		00h						
		usb:usb_inst state_c[20]		1h		4h				
-		usb:usb_inst cnt0[90]								

前面局部放大图

log:	Trig @ 3	2018/05/11 20:14:32 (0:0:1.3 elapsed)																	click t	o insert ti	me bar		
Туре	Alias	Name	-3	-2	-1	9	1	2	3	4	5		ì	7	8	9	1ρ	1,1	12	1,3	14	15	16
×.		flag_a																					
×.		flag_d																					
×.		sloe																					
-		slrd																					
ž		slwr																					
*																				2h			
1		usb_data[150]		FFFF	'n	X_	000h 🛛	0100h 🔾 🕻	200h	<u>озоон X</u>	0400h	0500h	0600	h X 0700	h) 080	00h (09	ooh 🛛 oA	ooh 🛛 ob	ooh 🗸 oC	ooh 🛛 oDe	ooh X oEc	oh 🕻 of	-00h 1
-		usb:usb_inst cmd_flag																					
6		⊡ usb:usb_inst cmd_data[150]		FFFF	'n		<u>000h X</u>	0100h 🛛 🕻	200h	<u>озоон X</u>	0400h	0500h	0600	h 🛛 0700	h) 080	00h 🛛 09	<u>ooh 🛛 oA</u>	ooh 🛛 ob	00h X 0C	ooh X oDo	ooh X oEc	oh X of	00h 1
					00h		Х	01h X	02h 🚶	озн Х	04h)	05h	06h	X 071) (oa	sh X o	9h 0	AhXo	Bh 🗸 O	<u>Ch 🛛 0[</u>	Dh 🚶 OE	ih (c	JFh (
-				1h																	4	h	
6																				0			
								4h															

nttps://blog.csdn.net/chengfengwenalan

后面局部放大图,注意只有在flag_d为高时,slwr为低才是有效写,否则就是无效写,因为当flag_d为低时,表示写满了,这时fifo就会弃之后的写数据了(因为已经写满了,也装不下了是吧,总不能硬塞是吧,哈哈~)

ing @	2010/05/1120.14.52 (0.0.1.5 clapsed)																area to ma	CAR SHITE D		
Alias	Name	241	242	243	244	245	246	247	248	249	250	251	252	253 2	54 28	52	56 2	§7 2§	8 3	259
	flag_a																			
	flag_d																Ц			
	sloe																			
	slrd																			
	slwr																h			
	. fifo_addr[10]																	2h		
	usb_data[150]	(F10	00h X F20	oh 🛛 F300	h X F40	oh 🛛 F500	h X F600	h (F700	h 🛛 F800	h X F900	h X FAOC	h X FBC	oh X FCoc	h X FDooh	FEOOh	FFooh	0000h	Х		
	usb:usb_inst cmd_flag																			
		(F10	00h X F20	oh 🛛 F300	h X F400	oh 🛛 F500	h X F600	h 🛛 F700	h 🛛 F800	h X F900	h X FAOC	h X FBC	oh X FCod	h X FDooh	FEOOh	FFooh	0000h	Х		
		(F1	1h 7 F2	n <mark>F3h</mark>		n <mark>F5h</mark>	F6h	F7h		F9t	X FAh	L X FE	h 🔪 FCh	X FDh	FEh	FFh	ooh	01h		
										4h								Х		
																		0		

PC端接受到的数据要2个字节一起读,因为usb是16位发送的,可以看出接受到的数据的确是0000-00FF

😙 EZ-USB Interface	– 🗆 X
Device EZ-USB Clear Load Mon S EEPROM Select Mon	
Get Dev Get Conf Get Pipes Get Strings Download Re-Load Lg EEPROM URB Stat HO	LD RUN
Vend Req Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT - Hex Bytes	C0 B4 04 81 00 01 00 👻
Iso Trans Pipe 1 Length 128 2 Packet Size Packets	
Bulk Trans Pip 2: Endpoint 6 IN v Length 512 Hex Bytes 5	•
Reset Pipe Abort Pipe File Trans. Pipe 2: Endpoint 6 IN	
Set IFace Interface 0 AttSetting 0	
Buffer Contents	^
0010 00 08 00 09 00 0A 00 UB 00 0C 00 0D 00 0E 00 0F	
0020 00 10 00 11 00 12 00 13 00 14 00 15 00 16 00 17	
0030 00 18 00 19 00 1A 00 1B 00 1C 00 1D 00 1E 00 1F	
0040 00 20 00 21 00 22 00 23 00 24 00 25 00 26 00 27	
0050 00 28 00 29 00 2A 00 2B 00 2C 00 2D 00 2E 00 2F	
0060 00 30 00 31 00 32 00 33 00 34 00 35 00 36 00 37	
0070 00 38 00 39 00 3A 00 3B 00 3C 00 3D 00 3E 00 3F	
0080 00 40 00 41 00 42 00 43 00 44 00 45 00 46 00 47	
0090 00 48 00 49 00 4A 00 4B 00 4C 00 4D 00 4E 00 4F	
100A0 00 50 00 51 00 52 00 53 00 54 00 55 00 56 00 57	
0000 00 50 00 53 00 54 00 58 00 50 00 50 00 52 00 5F	
https:///bl/og.esdramet/chd	ngreengwenallan.

注意: pc接受数据按我标的编码顺序执行

😙 EZ-	USE	3 Int	erfa	ace																				-	-			\times
Device	E	Z-L	JSB	_	_	_	_	_		•	ł	ର୍ଧୁ ପ	ear	Lo	oad N	lon	S EE	PROM	Se	ect M	on							
Get De	ev	G	et Co	nf	G	et Pip	es	Ge	t Stri	ngs	D	ownic	ad	R	le-Lo:	ad	Lg EE	PROM		RB St	at	но	DLD		RUN	4		
Vend R	leq	Re	q ()x00	Va	alue	OxC)000) In	dex	Ox	000	0	Leng	th [0	Di	r 0 (DUT	•	Hex	Byte	s C	0 B4	04 8	31 0	0 01	00 👻
lso Tra	ns	Pip	e [_		-			•	Ler	ngth	12	8	F	Packe	t Size			Pa	ickets	;	,	1				
Bulk Tra	ans	Pip	e 2	: Er	ndpo	oint	6 IN	ł		•	Ler	ngth	51	2		Hex	Bytes	5								•		
Reset P	Pipe	Ab	ort P	ipe	Fil	e Tra	ns		Pipe	2	En	dpa	int	6 IN		-												
Set IFa	ice	Int	erfa	ce 🛛	D	А	ltSet	ting	0																			
0100	00	80	00	81	00	82	00	83	00	84	00	85	00	86	00	87												^
0110	00	88	00	89	00	8A	00	8B	00	8C	00	8D	00	8E	00	8F												
0120	00	90	00	90	00	92	00	93	00	94	00	95	00	96	00	97												
0140	00	AO	00	A1	00	A2	00	A3	00	A4	00	AS	00	AG	00	A7												
0150	00	AS	00	Α9	00	AA	00	AB	00	AC	00	AD	00	AE	00	AF												
0160	00	B 0	00	B1	00	B2	00	В3	00	В4	00	B5	00	в€	00	B7												
0170	00	B8	00	В9	00	BA	00	BB	00	BC	00	BD	00	BE	00	BF												
0180	00	C0	00	C1	00	C2	00	C3	00	C4	00	C5	00	CE	00	C7												
0190	00	C8	00	C9	00	CA	00	CB	00	CC	00	CD	00	CE	00	CF												
01A0	00	DO	00	D1	00	D2	00	D3	00	D4	00	D5	00	DG	00	D7												
0180	00	D8	00	D9	00	DA	00	DB	00	DC	00	DD	00	DE	00	DF												
0100	00	E8	00	R9	00	RA	00	EB	00	EC	00	ED	00	RR	00	र म स स												
01E0	00	FO	00	F1	00	F2	00	F3	00	F4	00	F5	00	F6	00	F7												
01F0	00	F8	00	F9	00	FA	00	FB	00	FC	00	FD	00	FE	00	FF	′b1	og.	CSC	ln.	net	t/c	he	ng	fen	gw	ena	lan ,

下面轮到"读"出场了

trigge	er: 201	8/05/10 22:52:05 #0	Lock mode:	💕 Allow all chan	ges 🔹		
		Node	Data Enable	Trigger Enable	Trigger Conditions		
Туре	Alias	Name	61	61	1 🗹 Basic AND 🔹		
♦*		flag_a	\checkmark				
♦*		flag_d	\checkmark				
♦*		sloe	\checkmark				
♦*		slrd	\sim	✓	<u>0</u>		
*		slwr					
*		fifo_addr[10]			Xh		
1		[.] usb_data[150]			XXXXh		
*		usb:usb_inst cmd_flag					
-					XXXXh		
-					XXh		
-					Xh		
R C		[.] usb:usb_inst cnt0[90]	\checkmark		XXXXXXXXXXb		

https://blog.csdn.net/chengfengwenalan

pc发送数据按1-->2-->3的步骤,可以看出我们发送了12 34 56 78 这4个字节

😙 EZ-USB Interface —	×										
Device EZ-USB Clear Load Mon S EEPROM Select Mon											
Get Dev Get Conf Get Pipes Get Strings Download Re-Load Lg EEPROM URB Stat HOLD RUN											
Vend Req Req 0x00 Value 0x0000 Index 0x0000 Length 0 Dir 0 OUT - Hex Bytes C0 B4 04 81 00 01	00 🖵										
Iso Trans Pipe Length 128 Packet Size Rackets											
Bulk Trans Pip 2 0: Endpoint 2 OUT V Length 4 Hex Brites 12345678											
Reset Pipe Abort Pipe File Trans Pipe D: Endpoint 2 OUT V											
Set IFace Interface O AltSetting O											
Bulk OUT Transfer Bulk OUT success.	^										
Buffer Contents 0000 12 34 56 78											
https://blog.csdn.net/chengfengwena	lan,										

log:	Trig @ 3	2018/05/11 20:39:41 (0:0:7.4 elapsed)										click t	o insert time ba	ır		
Туре	Alias	Name	-4	-3	-2	-1	q		1	2	3		4	5	9	7
*		flag_a														
×.		flag_d														
4	sloe															
-	slrd															
-		slwr					_				_					
- 👑		fifo_addr[10]	2h				oh			х						
1			FFFFh				3412h X			7856h						
4		usb:usb_in <mark>it cmd_flag</mark>														
6					FFFFh			3412h	_X							7856h
ł													ooh			
-	Image: State_c[20]				1h				2h		Х					
-	usb:usb_inst cnt0[90]				c)			X OON		2					
								1								

注意这里我是设置了cmd_flag标志信号的,只有cmd_flag为高时的cmd_data的数据才是有效的,也就是pc发送 过来的数据。

这里顺带插一嘴:





In this example, the read port is operating at twice the frequency of the write port. Writing two 16-bit words to the FIFO buffer increases the wrusedw flag to two and the rusedw flag to four. Four 8-bit read operations empty the FIFO buffer. The read begins with the least-significant 8 bits from the 16-bit word written followed by the mostsignificant 8 bits.

Figure 8. Writing 8-Bit Words and Reading 16-Bit Words



alter 的fifo ip 是可以读写位宽不一致的,具体看下面的图



由上图可以看出这个和usb是一样的格式,都是先发低字节,然后再发高字节。或者说先接受低字节,然后再 接受高字节 至此本教程就全部介绍完了,讲的虽然有点简陋,但是基本的流程都已经讲到了,比起网上其他的教程还是要好不少的哈(自我陶醉一下),具体的看我的源代码。可以看出,使用cypress的usb还是比较简单的,因为usb协议什么的,他们都已经做好了,我们不需要考虑这些,要不然那一堆协议就看着头疼,具体的看我给的下载链接,我会把本教程所用到的工具,代码全部分享出来,也欢迎大家评论提问,不足之处还望指正~

六、福利

为了能及时回复大家,现在获取源码方式如下:

微信扫描下面的二维码关注【春哥笔记】公众号,回复"usb2"即可Get源码的获取方式:

