频率计实验报告

BobAnkh

1、实验目的：

频率计用于对一个未知频率的周期信号进行频率测量，在 1s 内对信号周期进行计数，即为此周期信号的频率。

2、设计方案：

按照实验指导书进行模块化设计，由于自身电脑性能等原因，出于仿真需要，已将所有信号的频率提高 1000 倍，在 1ms 内进行计数测量，后面报告内容与代码都已经是在该情况下的了。

频率量程模块设计利用外部信号同步产生其十分频信号，通过量程选择来选择输出原信号还是十分频之后的信号；系统时钟模块产生的是 1kHz 和 1MHz 的信号；控制信号产生模块输出计数允许信号（1 为允许计数）、清零信号（0 为清零）和锁存信号（1 为锁存，0 为透明传输）；计数器模块设计的是每 2ms 计数一次，即 1ms 计数和 1ms 清零，同时针对超过量程的信号保持显示最大量程（即超过 9999 的信号再未选用高量程模式下均在计数达到 9999 后保持显示 9999）；锁存器对应的就是 1ms 透明传输（即计数器正常计数时）和 1ms 锁存（即计数器清零时）；译码器使用扫频信号循环点亮四个数码管。这样设计可以看到的现象就是 1ms 示数不断增长后 1ms 示数保持，然后重新从 0 开始增长计数。

3、关键代码：

由于代码量较多，这里仅给出核心设计代码，其余设计代码、xdc 约束文件和 testbench 文件可以见对应附件。

计数器模块：

```
module counter(
input en,
input zero,
input signal,
output reg [3:0] thousand,
output reg [3:0] hundred,
output reg [3:0] ten,
output reg [3:0] one
    );
always @(negedge signal or negedge zero) begin
    if (zero==0) begin
        thousand <= 4'd0;
        hundred <= 4'd0;
        ten <= 4'd0;
        one <= 4'd0;
```

```
    end
    else begin
        if (en==0) begin
            thousand <= thousand;
            hundred <= hundred;
            ten <= ten;
            one <= one;
        end
        else begin
            one = one + 4'd1;
            if (one == 4'd10) begin
                one = 4'd0;
                ten = ten + 4'd1;
            end
            if (ten == 4'd10) begin
                ten = 4'd0;
                hundred = hundred + 4'd1;
            end
            if (hundred == 4'd10) begin
                hundred = 4'd0;
                thousand = thousand + 4'd1;
            end
            if (thousand == 4'd10) begin
                thousand = 4'd9;
                hundred = 4'd9;
                ten = 4'd9;
                one = 4'd9;
            end
        end
    end
end
endmodule
```

锁存器模块：

```
module save_16(
input [3:0] thousand,
input [3:0] hundred,
input [3:0] ten,
input [3:0] one,
input save,
output reg [3:0] thousand_o,
output reg [3:0] hundred_o,
output reg [3:0] ten_o,
```

```
output reg [3:0] one_o
    );
always @(*) begin
    if (~save) begin
        {thousand_o[3:0], hundred_o[3:0], ten_o[3:0], one_o[3:0]} <= {t
housand[3:0], hundred[3:0], ten[3:0], one[3:0]};
    end
    else begin
        thousand_o <= thousand_o;
        hundred_o <= hundred_o;
        ten_o <= ten_o;
        one_o <= one_o;
    end
end

endmodule
```

频率计整体：

```
module frequency(
input sigin,
input sysclk,
input modecontrol,
input rst,
output reg highfreq,
output [6:0] cathodes,
output [3:0] an
    );

wire signal_m;
wire clk_1;
wire clk_1k;
wire en;
wire zero;
wire save;
wire [3:0] thousand;
wire [3:0] hundred;
wire [3:0] ten;
wire [3:0] one;
wire [3:0] thousand_o;
wire [3:0] hundred_o;
wire [3:0] ten_o;
wire [3:0] one_o;

always @(*) begin
```

```
    highfreq <= modecontrol;
end

divide divide_feq(
.signal_1(sigin),
.rst(rst),
.select(modecontrol),
.signal_sel(signal_m)
);

clockgenerator system_clock(
.sysclk(sysclk),
.rst(rst),
.clk_1(clk_1),
.clk_1k(clk_1k)
);

control controller(
.clk(clk_1),
.rst(rst),
.en(en),
.zero(zero),
.save(save)
);

counter count(
.en(en),
.zero(zero),
.signal(signal_m),
.thousand(thousand),
.hundred(hundred),
.ten(ten),
.one(one)
);

save_16 saver(
.thousand(thousand),
.hundred(hundred),
.ten(ten),
.one(one),
.save(save),
.thousand_o(thousand_o),
.hundred_o(hundred_o),
.ten_o(ten_o),
```

```
.one_o(one_o)
);

decoder decode(
.thousand(thousand_o),
.hundred(hundred_o),
.ten(ten_o),
.one(one_o),
.clk(clk_1k),
.rst(rst),
.leds(cathodes),
.an(an)
);

endmodule
```

顶层文件：

```
module top(
input [1:0] testmode ,
input sysclk ,
input modecontrol,
input rst,
output highfreq,
output [6:0]cathodes,
output[3:0] an
);

wire sigin;

siginput signalin(
.testmode(testmode),
.sysclk(sysclk),
.rst(rst),
.sigin1(sigin)
);

frequency freq(
.sigin(sigin),
.sysclk(sysclk),
.modecontrol(modecontrol),
.rst(rst),
.highfreq(highfreq),
.cathodes(cathodes),
```

```
.an(an)
);

endmodule
```

4、文件清单：

在文件夹“频率计”中，在子文件夹“设计代码”中放置了时钟信号产生模块 clockgenerator.v，控制信号产生模块 control.v，计数模块 counter.v，译码模块 decoder.v，量程选择模块 divide.v，锁存器模块 save_16.v，频率计 frequency.v，测试信号模块 siginput.v 顶层文件 top.v，在子文件夹“testbench 代码和约束文件”中放置了 tb_top.v 和 freq.xdc

5、仿真结果及分析：

为了较好地测试设计功能，主要按照一下流程进行测试（已将测试也进行了 1000 倍映射）：

（1） 复位并设置部分初始数值
（2） 输入 3.125MHz 频率的信号，持续 3ms，量程选择原信号，观察结果
（3） 输入 6.250MHz 频率的信号，持续 2ms，量程选择原信号，观察结果
（4） 输入 0.050MHz 频率的信号，持续 2ms，量程选择原信号，观察结果
（5） 输入 12.500MHz 频率的信号，持续 2ms，量程选择原信号，观察结果
（6） 输入 12.500MHz 频率的信号，持续 2ms，量程选择高量程（即十分频），观察结果

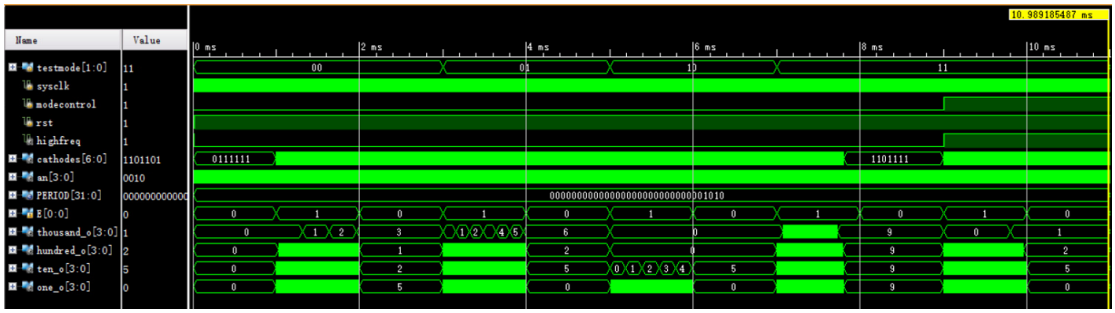
Testbench 部分关键代码如下：

```
initial
begin
    rst <= 1;
    sysclk <= 0;
    testmode <= 2'b00;
    modecontrol <= 0;
    #(PERIOD*2) rst <= 0;
    #(PERIOD*10) rst <= 1;
    #(PERIOD*300000) testmode <= 2'b01;
    #(PERIOD*200000) testmode <= 2'b10;
    #(PERIOD*200000) testmode <= 2'b11;
    #(PERIOD*200000) modecontrol <= 1;
    #(PERIOD*200000);
    $finish;
end
```

时序仿真：

为了更直观地检验，把锁存器的输出也添加到了波形中，与数码管输出同时观察来检验：

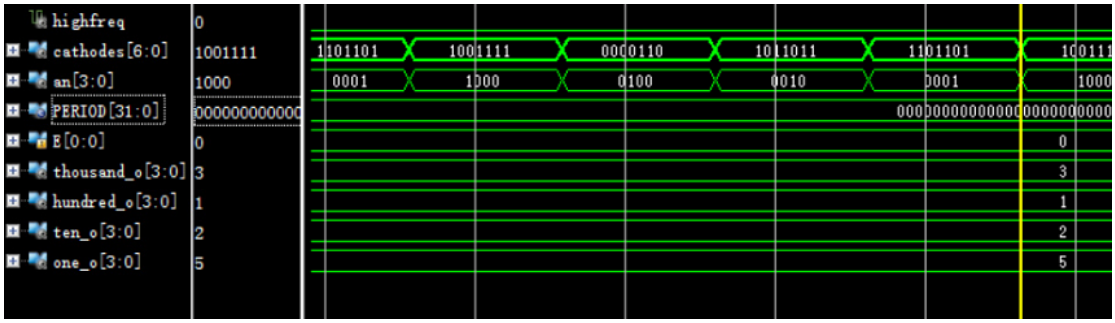
整体情况：



从这个整体图像上已经可以明显的看到，在使能信号为 1 时计数且锁存器透明传输，可以看到示数在上涨，在使能信号为 0 时锁存器锁存，可以清楚的读出对应各种输入情况的示数结果，原始量程的 3125,6250,50，而输入 12500 时选择原始量程就会计数达到 9999 后一直保持，选择高量程才会正常计数 1250，并且此时示意选择了高量程的 highfreq 电平会拉高。可以看到非常好地实现了预计的功能。

可以再来检验具体的数码管的输出是否正确，选取锁存阶段来查探：

（1）3125



可以看到循环点亮 4 个数码管，分别对应：

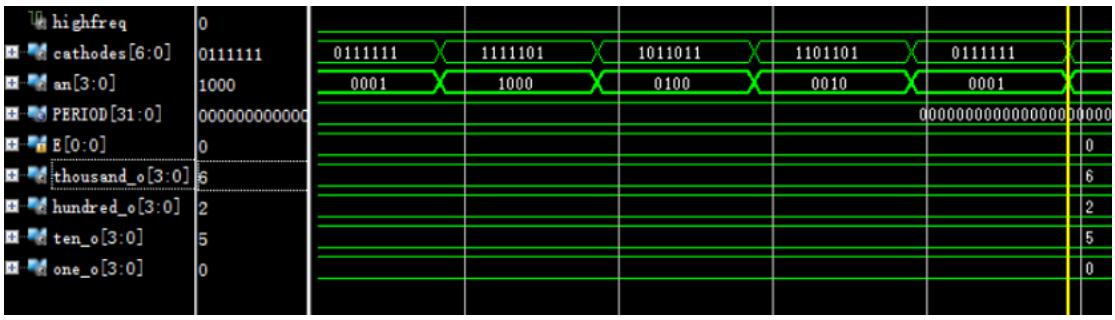
1001111->3

0000100->1

1011011->2

1101101->5

（2）6250



可以看到循环点亮 4 个数码管，分别对应：

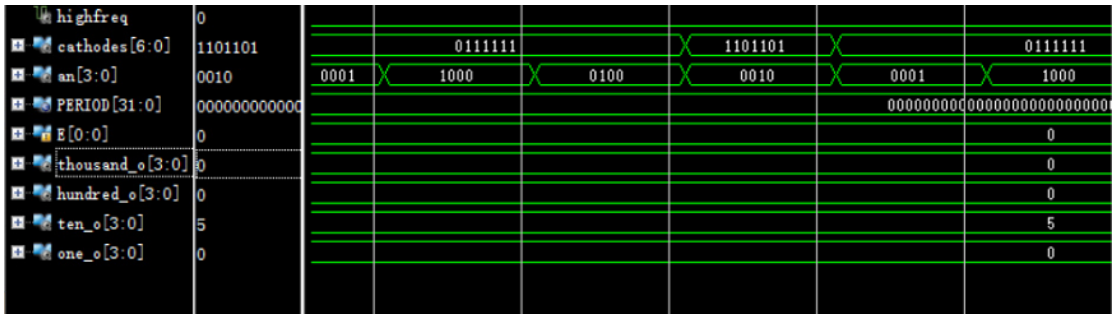
1111101->6

1011011->2

1101101->5

0111111->0

（3）50



可以看到循环点亮 4 个数码管，分别对应：

0111111->0

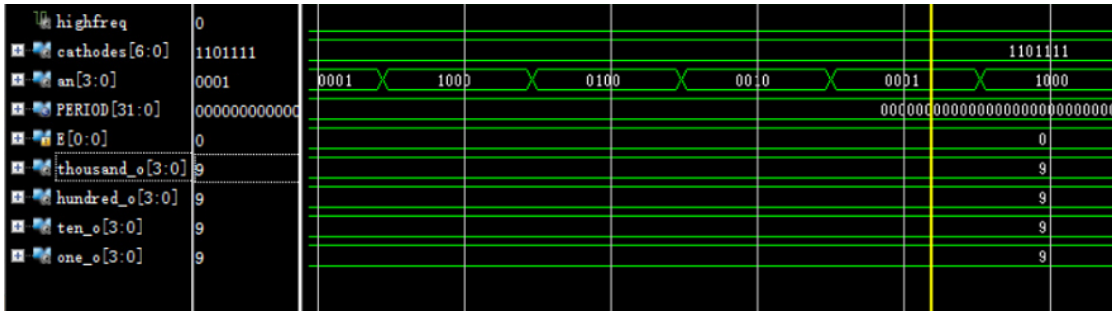
0111111->0

1101101->5

0111111->0

（4）12500

在选择原始量程情况下：



可以看到循环点亮 4 个数码管，分别对应：

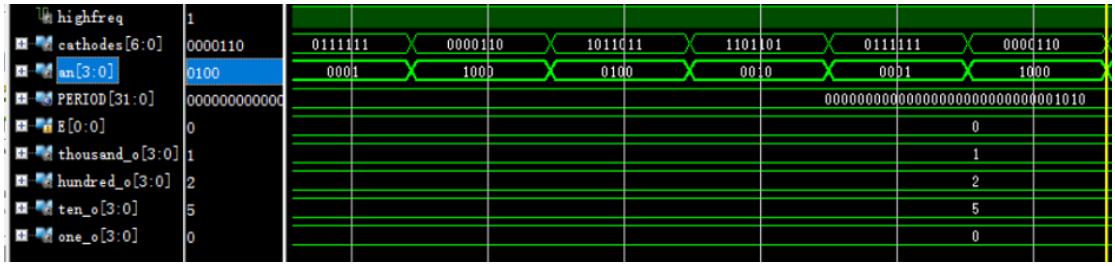
1101111->9

1101111->9

1101111->9

1101111->9

在选择高量程情况下：



可以看到循环点亮 4 个数码管，分别对应：

0000100->1

1011011->2

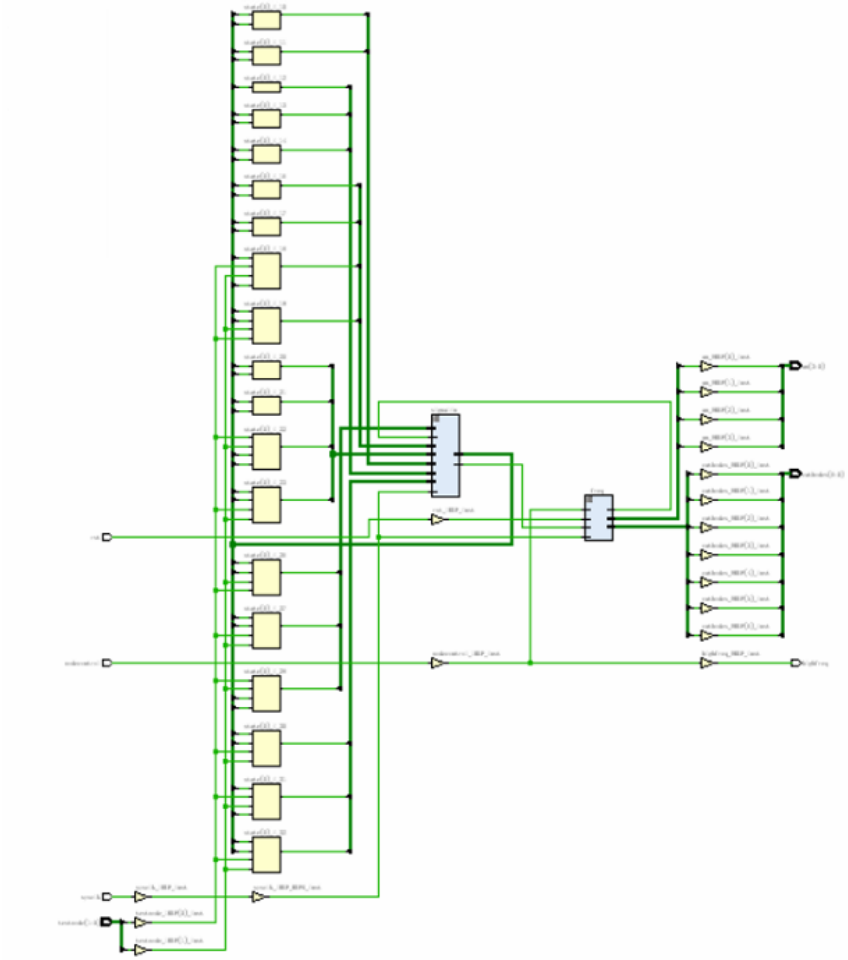
1101101->5

0111111->0

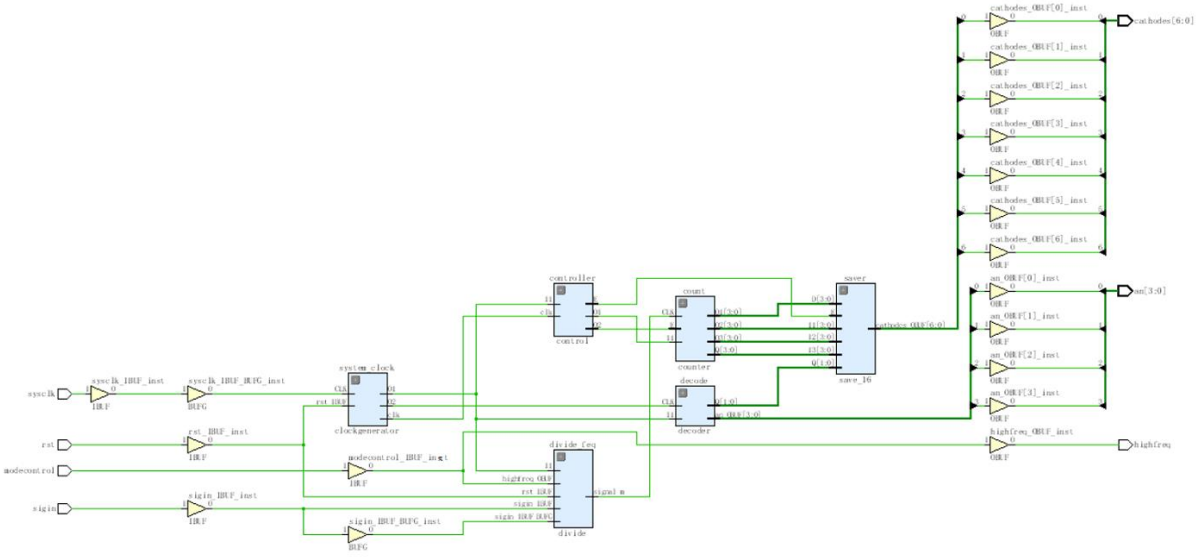
通过以上仿真内容可以说明对于频率计功能的良好实现，输出完全符合预期设计。

6、综合情况：

查看整体综合之后的电路原理图如下：



单独频率计模块综合的电路原理图如下：



FPGA 逻辑资源情况：

Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (210)	BUFGCTRL (32)
top	119	107	60	119	127	17	1
freq (frequency)	81	85	43	81	89	0	0
signalin (siginput)	28	22	17	28	28	0	0

可以看到，频率计模块使用了 81 个 LUT 和 85 个 Register

7、 时序性能

整体的时序性能：

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	3.271 ns	Worst Hold Slack (WHS):	0.245 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	66	Total Number of Endpoints:	66	Total Number of Endpoints:	67

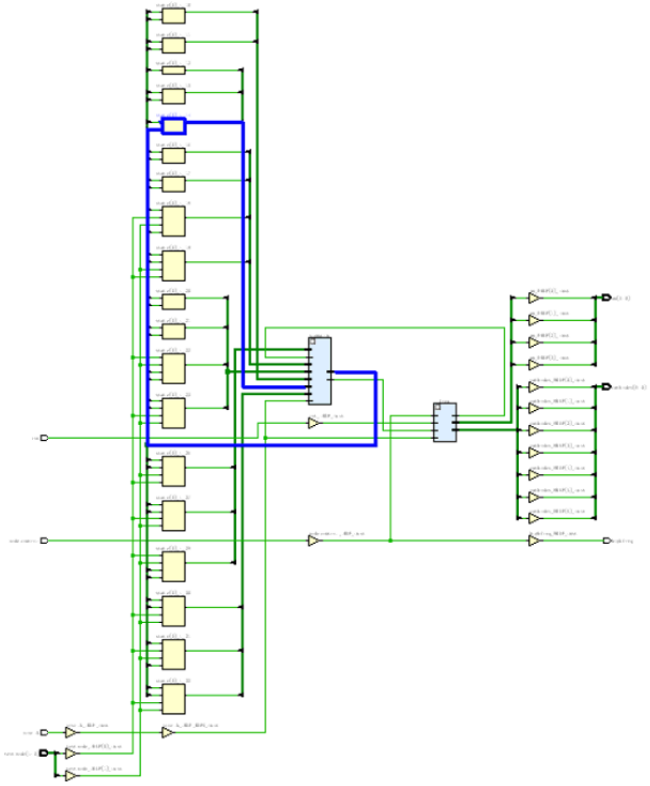
All user specified timing constraints are met.

可以看到，建立时间约束最坏的 slack 为 3.271ns，保持时间约束最坏的 slack 为 0.245ns

Intra-Clock Paths - CLK - Setup

Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Source Clock	Destination
Path 1	3.271	15	22	signalin/state_reg[1]/C	signalin/state_reg[20]/D	6.727	4.153	2.574	CLK	CLK
Path 2	3.274	14	22	signalin/state_reg[1]/C	signalin/state_reg[17]/D	6.724	4.150	2.574	CLK	CLK
Path 3	3.297	14	22	signalin/state_reg[1]/C	signalin/state_reg[19]/D	6.701	4.127	2.574	CLK	CLK
Path 4	3.369	14	22	signalin/state_reg[1]/C	signalin/state_reg[18]/D	6.629	4.055	2.574	CLK	CLK
Path 5	3.385	14	22	signalin/state_reg[1]/C	signalin/state_reg[16]/D	6.613	4.039	2.574	CLK	CLK
Path 6	3.389	13	22	signalin/state_reg[1]/C	signalin/state_reg[13]/D	6.610	4.036	2.574	CLK	CLK
Path 7	3.412	13	22	signalin/state_reg[1]/C	signalin/state_reg[15]/D	6.587	4.013	2.574	CLK	CLK
Path 8	3.484	13	22	signalin/state_reg[1]/C	signalin/state_reg[14]/D	6.515	3.941	2.574	CLK	CLK
Path 9	3.500	13	22	signalin/state_reg[1]/C	signalin/state_reg[12]/D	6.499	3.925	2.574	CLK	CLK
Path 10	3.504	12	22	signalin/state_reg[1]/C	signalin/state_reg[9]/D	6.496	3.922	2.574	CLK	CLK

仔细观察发现建立时间最短的十条路径都是输入测试信号的 sigin 模块的。在原理图上找出其中时间最短的一条路径：



查看它的具体信息如下：

Summary

Name	Path 1
Slack	3.271ns
Source	signalin/state_reg[1]/C (rising edge-triggered cell FDCE clocked by CLK {rise@0.000ns fall@5.000ns period=10.000ns})
Destination	signalin/state_reg[20]/D (rising edge-triggered cell FDCE clocked by CLK {rise@0.000ns fall@5.000ns period=10.000ns})
Path Group	CLK
Path Type	Setup (Max at Slow Process Corner)
Requirement	10.000ns (CLK rise@10.000ns - CLK rise@0.000ns)
Data Path Delay	6.727ns (logic 4.153ns (61.734%) route 2.574ns (38.266%))
Logic Levels	15 (CARRY4=12 LUT1=1 LUT2=2)
Clock Path Skew	-0.028ns
Clock Uncertainty	0.035ns

Source Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock CLK rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Site: P17	sysclk
net (fo=0)	0.000	0.000		sysclk
			Site: P17	sysclk_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.478	1.478	Site: P17	sysclk_IBUF_inst/O
net (fo=1, routed)	1.972	3.450		sysclk_IBUF
			Site: BUFGCTRL_X0Y0	sysclk_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.096	3.546	Site: BUFGCTRL_X0Y0	sysclk_IBUF_BUFG_inst/O
net (fo=66, routed)	1.636	5.182		signalin/sysclk_IBUF_BUFG
			Site: SLICE_X62Y9	signalin/state_reg[1]/C

Data Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
FDCE (Prop fdce C Q)	(f) 0.456	5.638	Site: SLICE_X62Y9	signalin/state_reg[1]/Q
net (fo=5, routed)	0.690	6.329		signalin/state_reg[1]
			Site: SLICE_X64Y9	signalin/state[0]_i_56/I0
LUT1 (Prop lut1 I0 O)	(r) 0.124	6.453	Site: SLICE_X64Y9	signalin/state[0]_i_56/O
net (fo=1, routed)	0.000	6.453		signalin/n_0_state[0]_i_56
			Site: SLICE_X64Y9	signalin/state_reg[0]_i_41/S[1]
CARRY4 (Prop carry4 S[1] CO[3])	(r) 0.533	6.986	Site: SLICE_X64Y9	signalin/state_reg[0]_i_41/CO[3]
net (fo=1, routed)	0.000	6.986		signalin/n_0_state_reg[0]_i_41
			Site: SLICE_X64Y10	signalin/state_reg[0]_i_40/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.117	7.103	Site: SLICE_X64Y10	signalin/state_reg[0]_i_40/CO[3]
net (fo=1, routed)	0.000	7.103		signalin/n_0_state_reg[0]_i_40
			Site: SLICE_X64Y11	signalin/state_reg[0]_i_35/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.117	7.220	Site: SLICE_X64Y11	signalin/state_reg[0]_i_35/CO[3]
net (fo=1, routed)	0.000	7.220		signalin/n_0_state_reg[0]_i_35
			Site: SLICE_X64Y12	signalin/state_reg[0]_i_34/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.117	7.337	Site: SLICE_X64Y12	signalin/state_reg[0]_i_34/CO[3]
net (fo=1, routed)	0.000	7.337		signalin/n_0_state_reg[0]_i_34
			Site: SLICE_X64Y13	signalin/state_reg[0]_i_24/CI
CARRY4 (Prop carry4 CI O[1])	(f) 0.323	7.660	Site: SLICE_X64Y13	signalin/state_reg[0]_i_24/O[1]
net (fo=2, routed)	0.803	8.463		state2[17]
			Site: SLICE_X63Y13	state[0]_i_14/I1
LUT2 (Prop lut2 I1 O)	(r) 0.306	8.769	Site: SLICE_X63Y13	state[0]_i_14/O
net (fo=1, routed)	0.000	8.769		signalin/I5[0]
			Site: SLICE_X63Y13	signalin/state_reg[0]_i_7/S[0]
CARRY4 (Prop carry4 S[0] CO[2])	(f) 0.536	9.305	Site: SLICE_X63Y13	signalin/state_reg[0]_i_7/CO[2]
net (fo=22, routed)	1.081	10.386		signalin/load
			Site: SLICE_X62Y9	signalin/state[0]_i_6/I1
LUT2 (Prop lut2 I1 O)	(r) 0.313	10.699	Site: SLICE_X62Y9	signalin/state[0]_i_6/O
net (fo=1, routed)	0.000	10.699		signalin/n_0_state[0]_i_6
			Site: SLICE_X62Y9	signalin/state_reg[0]_i_1/S[0]
CARRY4 (Prop carry4 S[0] CO[3])	(r) 0.532	11.231	Site: SLICE_X62Y9	signalin/state_reg[0]_i_1/CO[3]
net (fo=1, routed)	0.000	11.231		signalin/n_0_state_reg[0]_i_1
			Site: SLICE_X62Y10	signalin/state_reg[4]_i_1/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	11.345	Site: SLICE_X62Y10	signalin/state_reg[4]_i_1/CO[3]
net (fo=1, routed)	0.000	11.345		signalin/n_0_state_reg[4]_i_1
			Site: SLICE_X62Y11	signalin/state_reg[8]_i_1/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	11.459	Site: SLICE_X62Y11	signalin/state_reg[8]_i_1/CO[3]
net (fo=1, routed)	0.000	11.459		signalin/n_0_state_reg[8]_i_1
			Site: SLICE_X62Y12	signalin/state_reg[12]_i_1/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	11.573	Site: SLICE_X62Y12	signalin/state_reg[12]_i_1/CO[3]
net (fo=1, routed)	0.000	11.573		signalin/n_0_state_reg[12]_i_1
			Site: SLICE_X62Y13	signalin/state_reg[16]_i_1/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	11.687	Site: SLICE_X62Y13	signalin/state_reg[16]_i_1/CO[3]
net (fo=1, routed)	0.000	11.687		signalin/n_0_state_reg[16]_i_1
			Site: SLICE_X62Y14	signalin/state_reg[20]_i_1/CI
CARRY4 (Prop carry4 CI O[0])	(r) 0.223	11.910	Site: SLICE_X62Y14	signalin/state_reg[20]_i_1/O[0]
net (fo=1, routed)	0.000	11.910		signalin/n_7_state_reg[20]_i_1
FDCE			Site: SLICE_X62Y14	signalin/state_reg[20]/D
Arrival Time		11.910		

Destination Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock CLK rise edge)	(r) 10.000	10.000		
	(r) 0.000	10.000	Site: P17	sysclk
net (fo=0)	0.000	10.000		sysclk
			Site: P17	sysclk_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.408	11.408	Site: P17	sysclk_IBUF_inst/O
net (fo=1, routed)	1.868	13.276		sysclk_IBUF
			Site: BUFGCTRL_X0Y0	sysclk_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.091	13.367	Site: BUFGCTRL_X0Y0	sysclk_IBUF_BUFG_inst/O
net (fo=66, routed)	1.514	14.881		signalin/sysclk_IBUF_BUFG
			Site: SLICE_X62Y14	signalin/state_reg[20]/C
clock pessimism	0.273	15.154		
clock uncertainty	-0.035	15.119		
FDCE (Setup fdce C D)	0.062	15.181	Site: SLICE_X62Y14	signalin/state_reg[20]
Required Time		15.181		

由以上信息可以分析其 slack 是怎么得出的：数据通路上的到达时间 Arrival Time 为 11.910ns，又由其要求的时间 Required Time 为 15.181ns，由 Required Time 减去 Arrival Time 即可得到这条路径的建立时间的 slack，为 3.271ns

单独频率计的时序性能：

由于十条 slack 最短的路径都是由 siginput 模块带来的，我于是仅将频率计本身当做顶层模块，重新综合实现来看时序性能：

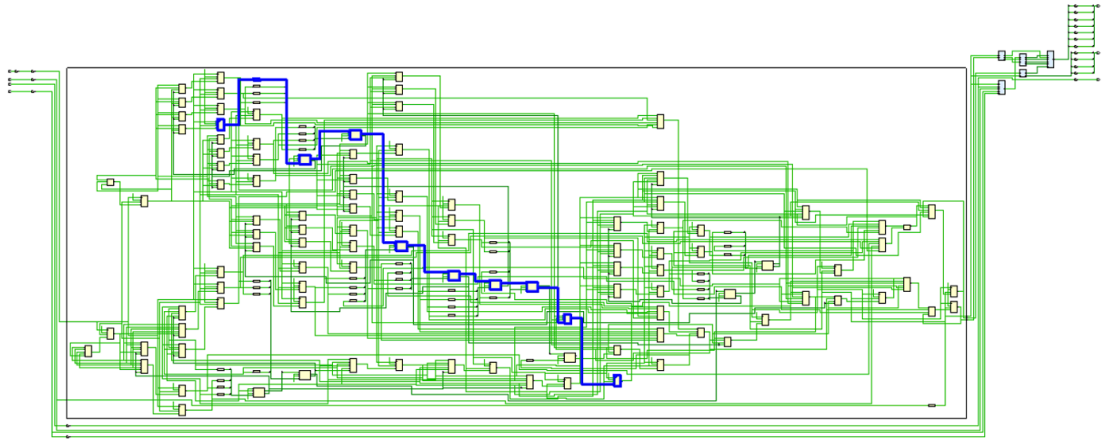
Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	6.589 ns	Worst Hold Slack (WHS):	0.265 ns	Worst Pulse Width Slack (WPWS):	4.460 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	44	Total Number of Endpoints:	44	Total Number of Endpoints:	45

All user specified timing constraints are met.

可以看到，建立时间约束最坏的 slack 为 6.589ns，保持时间约束最坏的 slack 为 0.265ns

Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Source Clock
Path 1	6.589	8	3	system_clock/count_1_reg[4]/C	system_clock/count_1_reg[22]/D	3.439	2.296	1.143	CLK
Path 2	6.703	7	3	system_clock/count_1_reg[4]/C	system_clock/count_1_reg[18]/D	3.325	2.182	1.143	CLK
Path 3	6.740	9	3	system_clock/count_1_reg[4]/C	system_clock/count_1_reg[25]/D	3.288	2.286	1.002	CLK
Path 4	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/clk_1_reg/D	3.232	1.039	2.193	CLK
Path 5	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/count_1_reg[0]/D	3.232	1.039	2.193	CLK
Path 6	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/count_1_reg[11]/D	3.232	1.039	2.193	CLK
Path 7	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/count_1_reg[12]/D	3.232	1.039	2.193	CLK
Path 8	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/count_1_reg[13]/D	3.232	1.039	2.193	CLK
Path 9	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/count_1_reg[1]/D	3.232	1.039	2.193	CLK
Path 10	6.755	3	27	system_clock/count_1_reg[24]/C	system_clock/count_1_reg[2]/D	3.232	1.039	2.193	CLK

仔细观察发现建立时间最短的十条路径都是系统时钟 clockgenerator 模块的。在原理图上找出其中时间最短的一条路径：



查看它的具体信息如下：

Summary	
Name	Path 1
Slack	6.589ns
Source	system_clock/count_1_reg[4]/C (rising edge-triggered cell FDCE clocked by CLK {rise@0.000ns fall@5.000ns period=10.000ns})
Destination	system_clock/count_1_reg[22]/D (rising edge-triggered cell FDCE clocked by CLK {rise@0.000ns fall@5.000ns period=10.000ns})
Path Group	CLK
Path Type	Setup (Max at Slow Process Corner)
Requirement	10.000ns (CLK rise@10.000ns - CLK rise@0.000ns)
Data Path Delay	3.439ns (logic 2.296ns (66.764%) route 1.143ns (33.236%))
Logic Levels	8 (CARRY4=6 LUT1=1 LUT4=1)
Clock Path Skew	-0.040ns
Clock Uncertainty	0.035ns

Source Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock CLK rise edge)	(r) 0.000	0.000		
	(r) 0.000	0.000	Site: P17	sysclk
net (fo=0)	0.000	0.000		sysclk
			Site: P17	sysclk_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.478	1.478	Site: P17	sysclk_IBUF_inst/O
net (fo=1, unplaced)	0.800	2.278		sysclk_IBUF
				sysclk_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.096	2.374		sysclk_IBUF_BUFG_inst/O
net (fo=44, unplaced)	0.800	3.174		system_clock/CLK
				system_clock/count_1_reg[4]/C

Data Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
FDCE (Prop fdce C Q)	(r) 0.496	3.670		system_clock/count_1_reg[4]/Q
net (fo=3, unplaced)	0.523	4.193		system_clock/count_1[4]
				system_clock/count_1[4]_i_3/I0
LUT1 (Prop lut1 I0 O)	(r) 0.295	4.488		system_clock/count_1[4]_i_3/O
net (fo=1, unplaced)	0.000	4.488		system_clock/n_0_count_1[4]_i_3
				system_clock/count_1_reg[4]_i_2/S[3]
CARRY4 (Prop carry4 S[3] CO[3])	(r) 0.401	4.889		system_clock/count_1_reg[4]_i_2/CO[3]
net (fo=1, unplaced)	0.009	4.898		system_clock/n_0_count_1_reg[4]_i_2
				system_clock/count_1_reg[8]_i_2/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	5.012		system_clock/count_1_reg[8]_i_2/CO[3]
net (fo=1, unplaced)	0.000	5.012		system_clock/n_0_count_1_reg[8]_i_2
				system_clock/count_1_reg[12]_i_2/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	5.126		system_clock/count_1_reg[12]_i_2/CO[3]
net (fo=1, unplaced)	0.000	5.126		system_clock/n_0_count_1_reg[12]_i_2
				system_clock/count_1_reg[16]_i_2/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	5.240		system_clock/count_1_reg[16]_i_2/CO[3]
net (fo=1, unplaced)	0.000	5.240		system_clock/n_0_count_1_reg[16]_i_2
				system_clock/count_1_reg[20]_i_2/CI
CARRY4 (Prop carry4 CI CO[3])	(r) 0.114	5.354		system_clock/count_1_reg[20]_i_2/CO[3]
net (fo=1, unplaced)	0.000	5.354		system_clock/n_0_count_1_reg[20]_i_2
				system_clock/count_1_reg[24]_i_2/CI
CARRY4 (Prop carry4 CI O[1])	(r) 0.348	5.702		system_clock/count_1_reg[24]_i_2/O[1]
net (fo=1, unplaced)	0.611	6.313		system_clock/data0[22]
				system_clock/count_1[22]_i_1/I3
LUT4 (Prop lut4 I3 O)	(r) 0.300	6.613		system_clock/count_1[22]_i_1/O
net (fo=1, unplaced)	0.000	6.613		system_clock/count_1_0[22]
FDCE				system_clock/count_1_reg[22]/D
Arrival Time		6.613		

Destination Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock CLK rise edge)	(r) 10.000	10.000		
	(r) 0.000	10.000	Site: P17	sysclk
net (fo=0)	0.000	10.000		sysclk
			Site: P17	sysclk_IBUF_inst/I
IBUF (Prop ibuf I O)	(r) 1.408	11.408	Site: P17	sysclk_IBUF_inst/O
net (fo=1, unplaced)	0.760	12.168		sysclk_IBUF
				sysclk_IBUF_BUFG_inst/I
BUFG (Prop bufg I O)	(r) 0.091	12.259		sysclk_IBUF_BUFG_inst/O
net (fo=44, unplaced)	0.760	13.019		system_clock/CLK
				system_clock/count_1_reg[22]/C
clock pessimism	0.116	13.134		
clock uncertainty	-0.035	13.099		
FDCE (Setup fdce C D)	0.103	13.202		system_clock/count_1_reg[22]
Required Time		13.202		

由以上信息可以分析其 slack 是怎么得出的：数据通路上的到达时间 Arrival Time 为 6.613ns，又由其要求的时间 Required Time 为 13.202ns，由 Required Time 减去 Arrival Time 即可得到这条路径的建立时间的 slack，为 6.589ns