数字逻辑电路大型实验报告

姓名 凌智城

学号 201806061211

指导教师 李如春

专业班级 通信工程 1803 班

学　　院 信息工程学院

提交日期 2020 年 7 月 14 日

一、实验目的

学习用 FPGA 实现数字系统的方法

二、实验内容

1．FPGA, Quartus II 和 VHDL 使用练习

2．四位数字频率计的设计

3. 信号发生器

三、四位数字频率计的设计

1．工作原理

频率就是周期性信号在单位时间（1s）内的变化次数。若在 1s 的时间间隔内测得这个周期性信号的重复变化次数为 N，则其频率 f 可表示为

$$f=N$$

由此可见，只要将被测信号作为计数器的时钟输入，让计数器从零开始计数，计数器计数 1s 后得到的计数值就是被测信号的频率值。利用上述电路可以得到突入 8.2-1 所示的数字频率计原理框图。控制电路首先给出清零信号，使计数器清零。然后闸门信号置为高电平，闸门开通，被测信号通过闸门送到计数器，计数器开始计数，1s 后，将闸门信号置为低电平，计数器停止计数，此时计数器的计数值就是被测信号频率。如果将计数值直接送到显示电路显示，那么在整个计数过程中，显示值将不断变化，无法看清显示值。在计数器和显示电路之间加了锁存器之后，控制器在闸门关闭后给出一锁存信号，将计数值存入锁存器，显示电路根据锁存器的输出显示频率值。这样，每测量一次频率值，显示值刷新一次。图 8.2-2 给出了数字频率计各信号的时序关系。

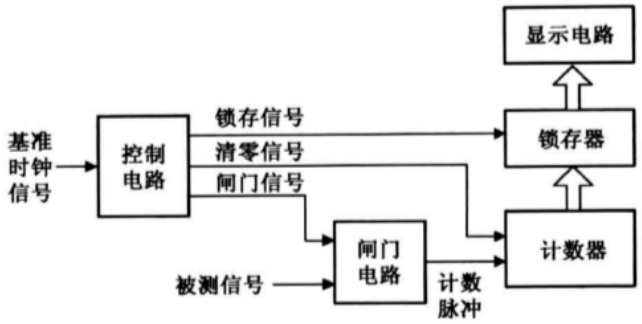


图 8.2-1　数字频率计原理框图

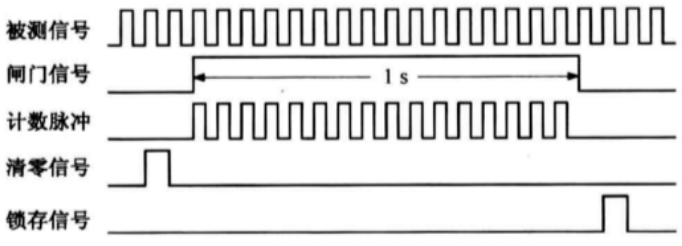


图 8.2-2　频率计控制信号时序图

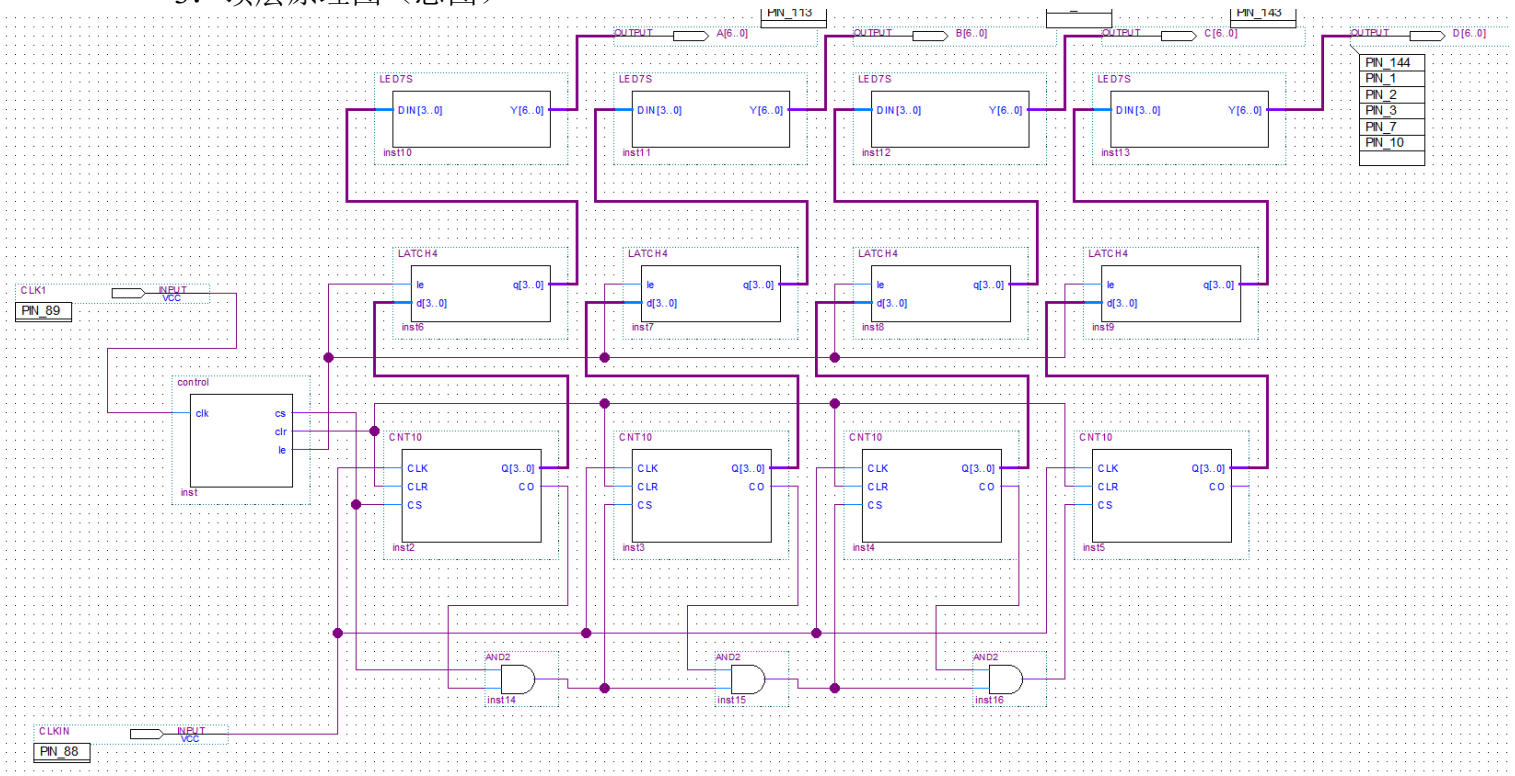
数字频率计是一个典型的数字系统，控制器构成控制单元，计数器和锁存器等构成数据处理单元，数据处理单元各模块逻辑功能比较简单，通常有标准的模块可供选择，而控制器一般需要自行设计。控制器除了基准时钟信号外，没有其他输入信号，因此，可采用教材例 4.5-1 介绍的序列信号发生器实现，当 CP 脉冲采用频率为 8Hz 的基准时钟是，则闸门信号的脉冲宽度刚好为 1s。

数字频率计的硬件电路主体部分有 FPGA 实现，在此基础上扩展 LED 显示电路及时钟电路即可。由于数字频率计的测频范围为 0~9999Hz，因此显示电路可采用 4 位 7 端 LED 数码管。始终电路用于产生 8Hz 基准时钟 CLK1，同时产生一路频率可变的时钟信号 CLKIN 作为作为数字频率计的被测信号，以方便频率计的测试。7 段 LED 数码管直接采用 EDA 实验板上的数码管，而时钟电路则由实验板上的晶体振荡器产生。

2．设计方案

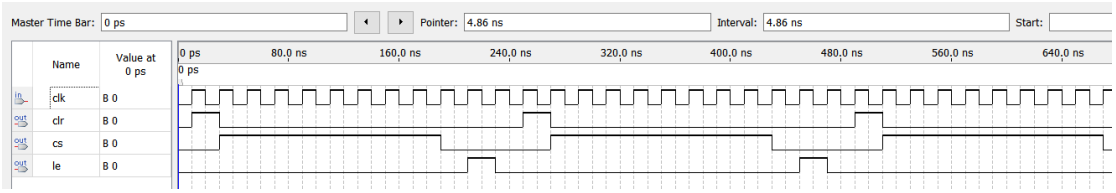
数字频率计主体部分采用 FPGA 实现，采用“自顶向下”的设计方法。先进行顶层设计，后进行底层模块设计。与计算机软件程序类似，顶层设计相当于主程序，底层模块相当于子程序。顶层设计可以采用原理图，也可以采用 VHDL 语言描述。采用原理图设计比较直观，但移植性较差。在以下的数字频率设计中，顶层设计采用原理图，底层模块采用 VHDL 语言描述。

3．顶层原理图（总图）

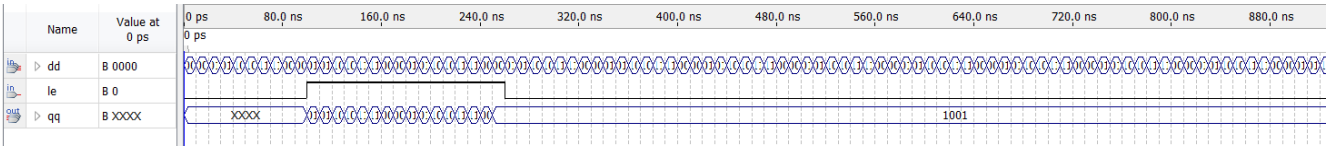


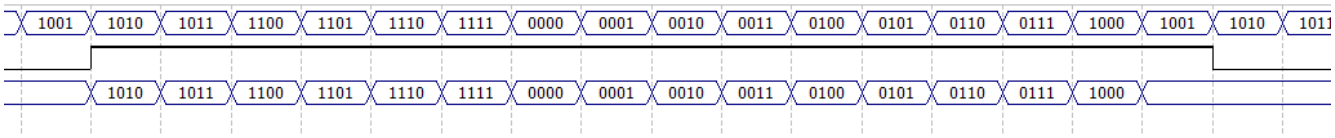
4．底层 4 个模块（控制信号产生模块，十进制计数器模块，锁存器模块，译码模块）的仿真结果。

Control

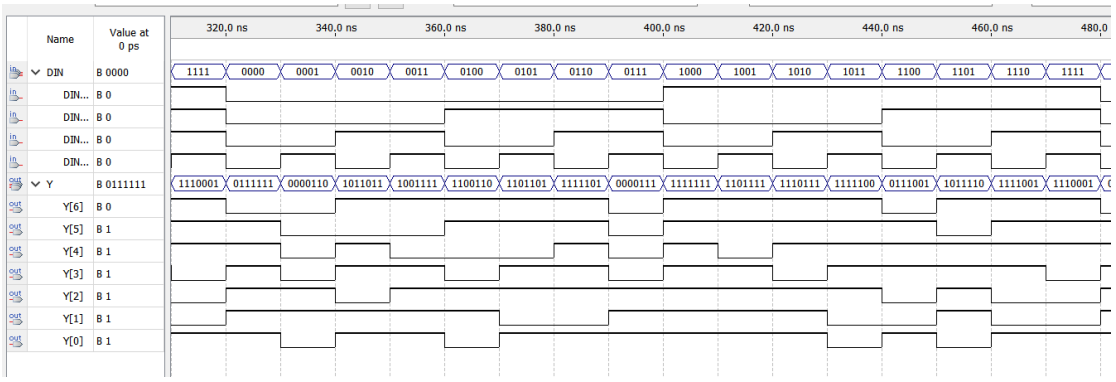


LATCH4

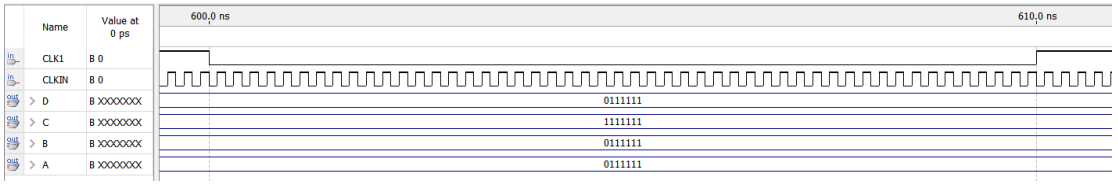




LED7S



总



5．项目处理

（1）器件选定

选择四个 7 段 LED 数码管 A、B、C、D 和时钟的 CLK1、CLKIN

（2）管脚锁定

D[6]	Output	PIN_11
D[5]	Output	PIN_10
D[4]	Output	PIN_7
D[3]	Output	PIN_3
D[2]	Output	PIN_2
D[1]	Output	PIN_1
D[0]	Output	PIN_144
C[6]	Output	PIN_143
C[5]	Output	PIN_142
C[4]	Output	PIN_141
C[3]	Output	PIN_138
C[2]	Output	PIN_137
C[1]	Output	PIN_136
C[0]	Output	PIN_135
CLKIN	Input	PIN_88
CLK1	Input	PIN_89
B[6]	Output	PIN_133
B[5]	Output	PIN_132
B[4]	Output	PIN_129
B[3]	Output	PIN_128
B[2]	Output	PIN_127
B[1]	Output	PIN_126
B[0]	Output	PIN_125
A[6]	Output	PIN_124
A[5]	Output	PIN_121
A[4]	Output	PIN_120
A[3]	Output	PIN_119
A[2]	Output	PIN_115
A[1]	Output	PIN_114
A[0]	Output	PIN_113
<<new node>>		

（3）编程下载

在编程窗口中单机左上方的“Hardware Setup”按钮，在打开的窗口中选择“USB-Blaster”。在编程模式“Mode”中选择“JTAG”，并用鼠标选中Program/Configure 下方的小方框。单机“Start”按钮，即开始下载操作，当“Progress”显示 100%时，下载结束。手动拔插低频晶体振荡器，产生不同的时钟信号 CLKIN（4Hz、32Hz、128Hz、512Hz、2048Hz），数码管上显示相同的频率数值，仿真结果正确无误。

6. 频率测量结果

被测频率（Hz）	实测频率（Hz）
4	4
64	64
128	128
512	512
2048	2048

四、信号发生器的设计

1．工作原理

5 分频电路实现从 25MHz 到 5MHz 的转变，通过设置内部锁相环 PLL 的分频因子和倍频因子得到 20MHz 的信号，正弦波、三角波、任意波的数据表 ROM 以及 D\A 转换器实现输出最终的信号。

2．设计方案

i. 锯齿波信号发生器

输入 CLK0 频率为 25MHz，经过 5 分频电路转化为 5MHz 的信号 CLK1，再经过八位二进制计数器以及 D/A 转换器输出锯齿波，了满足 D/A 转换器的时序要求，CLK1 经过反相以后再送 D/A。

CNT5

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity CNT5 is
	port(CLK: in std_logic;
		CO: out std_logic);
end CNT5;


architecture one of CNT5 is
signal Q:std_logic_vector(2 downto 0);
	begin
		process(CLK)
			begin

				if(CLK'EVENT and CLK = '1') then

					if(Q = 4) then
						Q <= "000";
					else
						Q <= Q + 1;
					end if;

				end if;
			end process;
		process (Q)
		begin
			if(Q = 4) then
				CO <= '1';
			else
				CO <= '0';
			end if;
		end process;
	end;
```

CNT8

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
```

```
entity CNT8 is
    port(CLK, CLR,CS: in std_logic;
         Q: buffer std_logic_vector(7 downto 0);
         CO: out std_logic);
end CNT8;


architecture one of CNT8 is
    begin
        process(CLK,CLR,CS)
            begin
                if(CLR = '1') then
                    Q <= "00000000";  --异步清零
                elsif(CLK'EVENT and CLK = '1') then
                    if(CS = '1') then
                        if(Q = 255) then
                            Q <= "00000000";
                        else
                            Q <= Q + 1;
                        end if;
                    end if;
                end if;
            end process;
        process (Q)
        begin
            if(Q = 255) then
                CO <= '1';
            else
                CO <= '0';
            end if;
        end process;
    end;
```

ii. 正弦信号发生器

外部时钟 CLK0 频率为 25MHz，通过 FPGA 内部锁相环 PLL 得到频率为 20MHz 的时 钟信号 CLK1。8 位二进制计数器（地址计数器）在时钟信号 CLK1 的作用下输出 8 位地址 信号。256 字节的正弦数据存放在 ROM 中，根据地址计数器输出的地址，将对应存储单元 的数据送高速 D/A 转换器。时钟 CLK1 不但作为地址计数器的同步时钟，也作为高速 D/A 数据寄存器的时钟信号。为了满足 D/A 转换器的时序要求，CLK1 经过反相以后再送 D/A 转换器。

iii. DDS 正弦信号发生器

利用 DDS 技术实现输出正弦信号频率步进可调，通过按键 KEY0 实现输出正弦信号频率从 1kHz、4kHz、…、31kHz 变化，输出频率由两位 LED 数码管显示。

CONTROL

```
LIBRARY IEEE;
```

```
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;

ENTITY control IS
    PORT(keycnt:IN STD_LOGIC;
         CLK: IN STD_LOGIC;
         control:OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
         LEDA,LEDB:OUT STD_LOGIC_VECTOR(6 DOWNTO 0)
         );
END control;

ARCHITECTURE one OF control IS
    SIGNAL q:STD_LOGIC_VECTOR(4 DOWNTO 0);

BEGIN
    PROCESS(keycnt)
    BEGIN
        IF(keycnt'EVENT AND keycnt='0')THEN
            IF(q=30)THEN
                q<="00000";
            ELSE
                q<=q+3 after 50ms;
            END IF;
        END IF;
    END PROCESS;
    PROCESS(q)
    BEGIN
        CASE q IS
            WHEN"00000"=>
            control<=X"00029F17";LEDA<="0000110";LEDB<="0111111";
            WHEN"00001"=>
            control<=X"00053E2E";LEDA<="1011011";LEDB<="0111111";
            WHEN"00010"=>
            control<=X"0007DD45";LEDA<="1001111";LEDB<="0111111";
            WHEN"00011"=>
            control<=X"000A7C5C";LEDA<="1100110";LEDB<="0111111";
            WHEN"00100"=>
            control<=X"000D1B71";LEDA<="1101101";LEDB<="0111111";
            WHEN"00101"=>
            control<=X"000FBA8A";LEDA<="1111101";LEDB<="0111111";
            WHEN"00110"=>
            control<=X"001259A1";LEDA<="0000111";LEDB<="0111111";
            WHEN"00111"=>
            control<=X"0014F8B8";LEDA<="1111111";LEDB<="0111111";
            WHEN"01000"=>
            control<=X"001797CF";LEDA<="1101111";LEDB<="0111111";
            WHEN"01001"=>
            control<=X"001A36E6";LEDA<="0111111";LEDB<="0000110";
            WHEN"01010"=>
```

```
                control<=X"001CD5FD";LEDA<="0000110";LEDB<="0000110";
                WHEN"01011"=>
                control<=X"001F7514";LEDA<="1011011";LEDB<="0000110";
                WHEN"01100"=>
                control<=X"0022142B";LEDA<="1001111";LEDB<="0000110";
                WHEN"01101"=>
                control<=X"0024B342";LEDA<="1100110";LEDB<="0000110";
                WHEN"01110"=>
                control<=X"00275259";LEDA<="1101101";LEDB<="0000110";
                WHEN"01111"=>
                control<=X"0029F170";LEDA<="1111101";LEDB<="0000110";
                WHEN"10000"=>
                control<=X"002C9087";LEDA<="0000111";LEDB<="0000110";
                WHEN"10001"=>
                control<=X"002F2F9E";LEDA<="1111111";LEDB<="0000110";
                WHEN"10010"=>
                control<=X"0031CEB5";LEDA<="1101111";LEDB<="0000110";
                WHEN"10011"=>
                control<=X"00346DCC";LEDA<="0111111";LEDB<="1011011";
                WHEN"10100"=>
                control<=X"00370CE3";LEDA<="0000110";LEDB<="1011011";
                WHEN"10101"=>
                control<=X"0039ABFA";LEDA<="1011011";LEDB<="1011011";
                WHEN"10110"=>
                control<=X"003C4B11";LEDA<="1001111";LEDB<="1011011";
                WHEN"10111"=>
                control<=X"003EEA28";LEDA<="1100110";LEDB<="1011011";
                WHEN"11000"=>
                control<=X"0041893F";LEDA<="1101101";LEDB<="1011011";
                WHEN"11001"=>
                control<=X"00442856";LEDA<="1111101";LEDB<="1011011";
                WHEN"11010"=>
                control<=X"0046C76D";LEDA<="0000111";LEDB<="1011011";
                WHEN"11011"=>
                control<=X"00496684";LEDA<="1111111";LEDB<="1011011";
                WHEN"11100"=>
                control<=X"004C059B";LEDA<="1101111";LEDB<="1011011";
                WHEN"11101"=>
                control<=X"004EA4B2";LEDA<="0111111";LEDB<="1001111";
                WHEN"11110"=>
                control<=X"005143C9";LEDA<="0000110";LEDB<="1001111";
                WHEN"11111"=>
                control<=X"0053E2E0";LEDA<="1011011";LEDB<="1001111";
                WHEN OTHERS=>
                control<=X"00029F17";LEDA<="0000110";LEDB<="0111111";
            END CASE;
        END PROCESS;
END one;
```

PHASE_ACC

```
library ieee;
```

```
use ieee.std_logic_1164.all;

use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity phase_acc is
    port(
            clk:in std_logic;
            freqin:in std_logic_vector(31 downto 0);
            romaddr:out std_logic_vector(7 downto 0));
end phase_acc;

architecture one of phase_acc is
        signal acc:std_logic_vector(31 downto 0);
begin
    process(clk)
    begin
        if(clk'event and clk='1')then
            acc<=acc-freqin;
            end if;
        end process;
        romaddr<=acc(31 downto 24);
end one;
```

iv. DDS 任意波形信号发生器

在 DDS 正弦信号发生器的基础上增加一个四选一数据选择器，通过按键 SW0 和 SW1 实现三种波形的选择输出，同时也可以通过按键 KEY0 实现信号的频率可调。

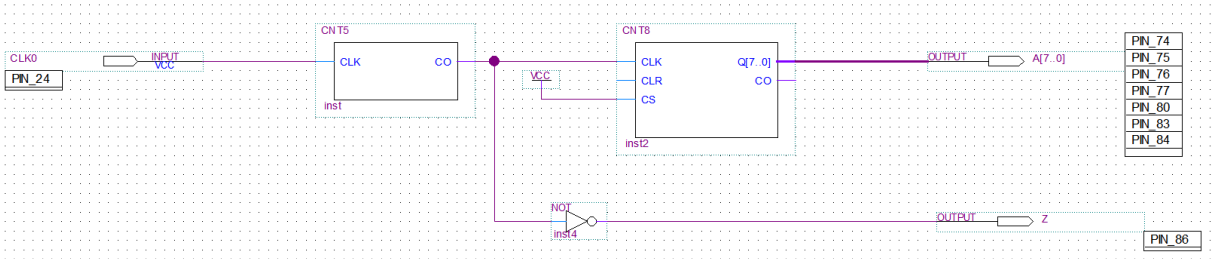
四选一数据选择器

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;

ENTITY CONTROL IS
    PORT(SW0,SW1:IN STD_LOGIC;
          IN1,IN2,IN3:IN STD_LOGIC_VECTOR(7 DOWNTO 0);
          OU:OUT STD_LOGIC_VECTOR(7 DOWNTO 0)
          );
END CONTROL;

ARCHITECTURE one OF CONTROL IS
BEGIN
    PROCESS(SW0,SW1)
    BEGIN
        IF(SW0='0')THEN
            IF(SW1='0')THEN
                OU<=IN1;
            ELSE
```

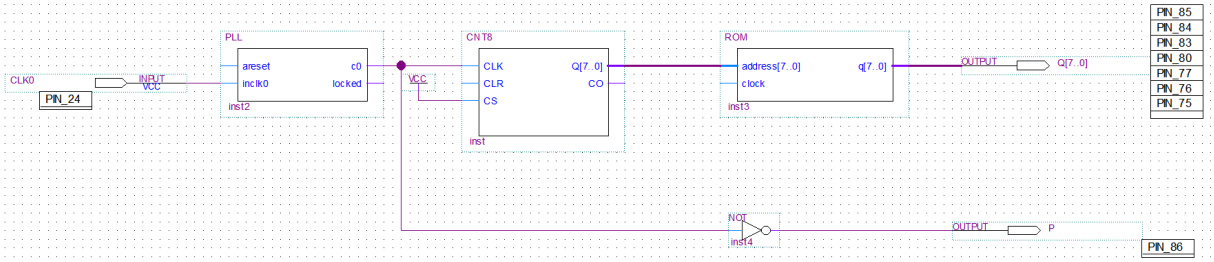
```
                OU<=IN1;
            END IF;
        END IF;
        IF(SW0='1')THEN
            IF(SW1='0')THEN
                OU<=IN2;
            ELSE
                OU<=IN3;
            END IF;
        END IF;
    END PROCESS;
END one;
```

3．顶层原理图（总图）

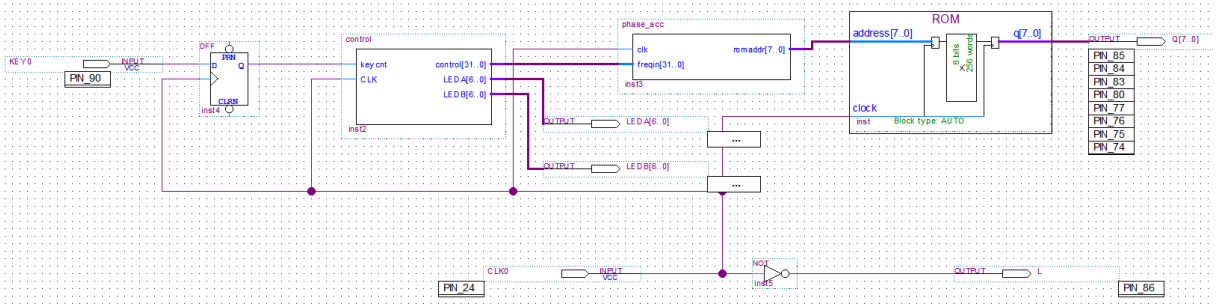
i. 锯齿波信号发生器



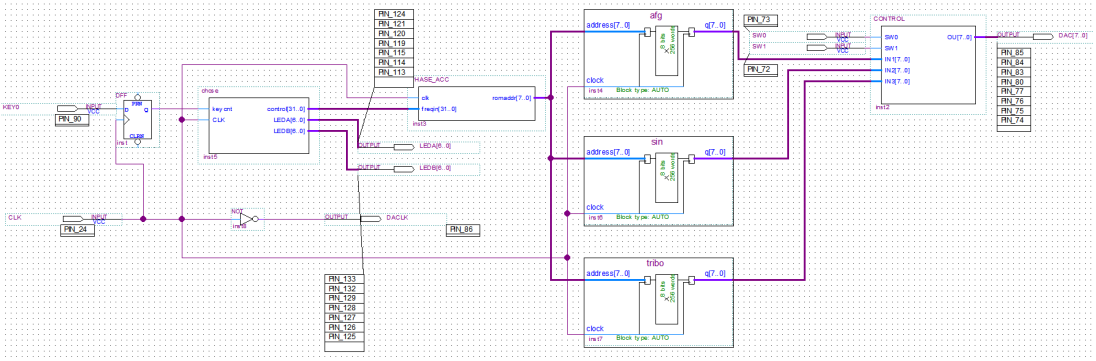
ii. 正弦信号发生器



iii. DDS 正弦信号发生器



iv. DDS 任意波形信号发生器



4．仿真结果（可选）。

5．项目处理

（1）器件选定

i. 锯齿波信号发生器

时钟 CLK0，高速 D\A 转换器 A

ii. 正弦信号发生器

时钟 CLK0，高速 D\A 转换器 Q

iii. DDS 正弦信号发生器

高速 D\A 转换器 Q，按钮 KEY0，时钟 CLK0，两只七段 LED 数码管 LEDA 和 LEDB

iv. DDS 任意波形信号发生器

按钮 KEY0，高速 D\A 转换器，时钟 CLK0，两只七段 LED 数码管 LEDA 和 LEDB，电平开关 SW0、SW1

（2）管脚锁定

i. 锯齿波信号发生器

Node Name	Direction	Location	I/O Bank
A[7]	Output	PIN_85	5
A[6]	Output	PIN_84	5
A[5]	Output	PIN_83	5
A[4]	Output	PIN_80	5
A[3]	Output	PIN_77	5
A[2]	Output	PIN_76	5
A[1]	Output	PIN_75	5
A[0]	Output	PIN_74	5
CLK0	Input	PIN_24	2
Z	Output	PIN_86	5
<<new node>>			

ii. 正弦信号发生器

Node Name	Direction	Location	I/O Bank
CLK0	Input	PIN_24	2
P	Output	PIN_86	5
Q[7]	Output	PIN_85	5
Q[6]	Output	PIN_84	5
Q[5]	Output	PIN_83	5
Q[4]	Output	PIN_80	5
Q[3]	Output	PIN_77	5
Q[2]	Output	PIN_76	5
Q[1]	Output	PIN_75	5
Q[0]	Output	PIN_74	5
<<new node>>			

iii. DDS 正弦信号发生器

Node Name	Direction	Location	I/O Bank
CLK0	Input	PIN_24	2
KEY0	Input	PIN_90	6
L	Output	PIN_86	5
LEDA[6]	Output	PIN_124	7
LEDA[5]	Output	PIN_121	7
LEDA[4]	Output	PIN_120	7
LEDA[3]	Output	PIN_119	7
LEDA[2]	Output	PIN_115	7
LEDA[1]	Output	PIN_114	7
LEDA[0]	Output	PIN_113	7
LEDB[6]	Output	PIN_133	8
LEDB[5]	Output	PIN_132	8
LEDB[4]	Output	PIN_129	8
LEDB[3]	Output	PIN_128	8
LEDB[2]	Output	PIN_127	7
LEDB[1]	Output	PIN_126	7
LEDB[0]	Output	PIN_125	7
Q[7]	Output	PIN_85	5
Q[6]	Output	PIN_84	5
Q[5]	Output	PIN_83	5
Q[4]	Output	PIN_80	5
Q[3]	Output	PIN_77	5
Q[2]	Output	PIN_76	5
Q[1]	Output	PIN_75	5
Q[0]	Output	PIN_74	5
<<new node>>			

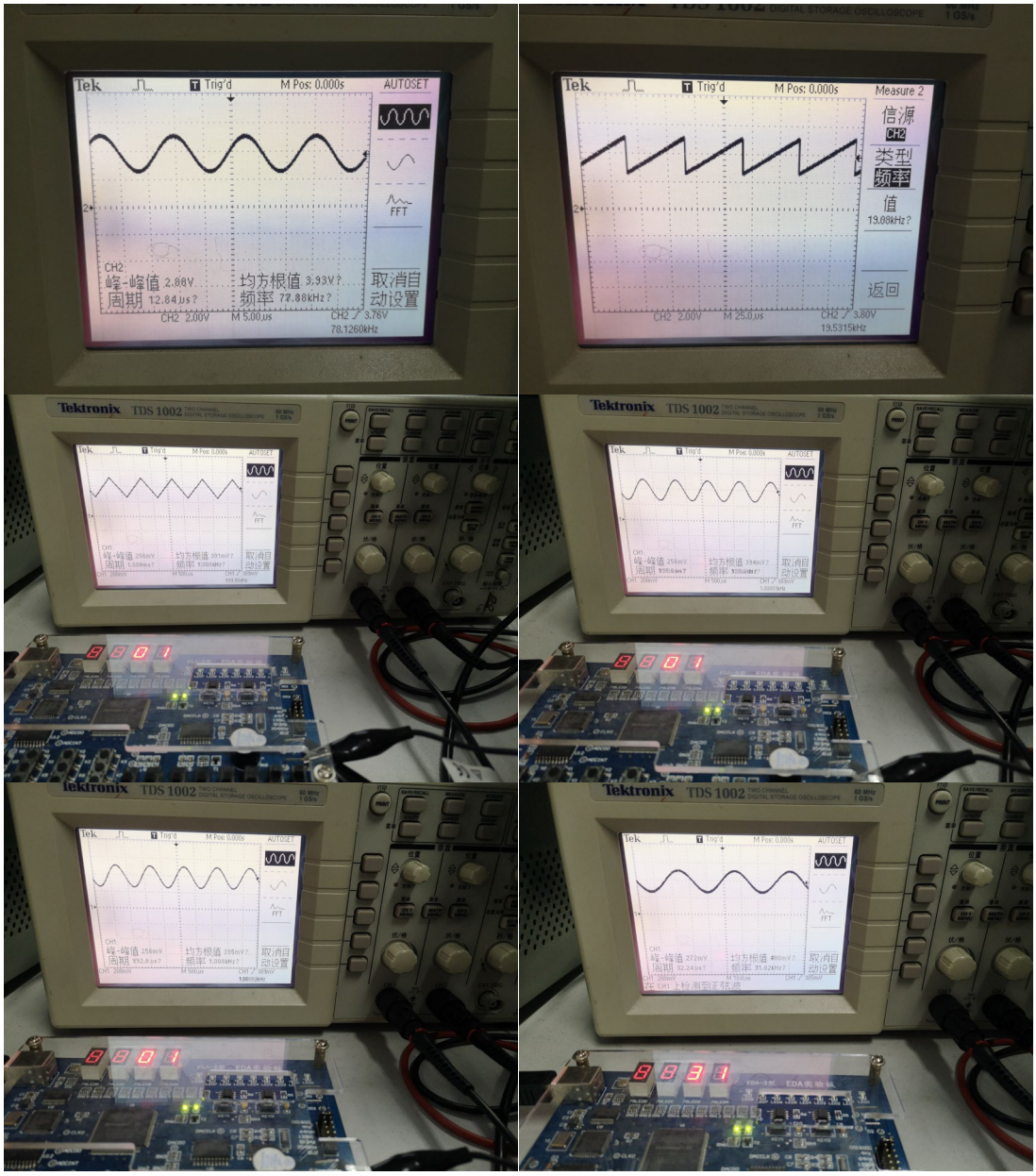
iv. DDS 任意波形信号发生器

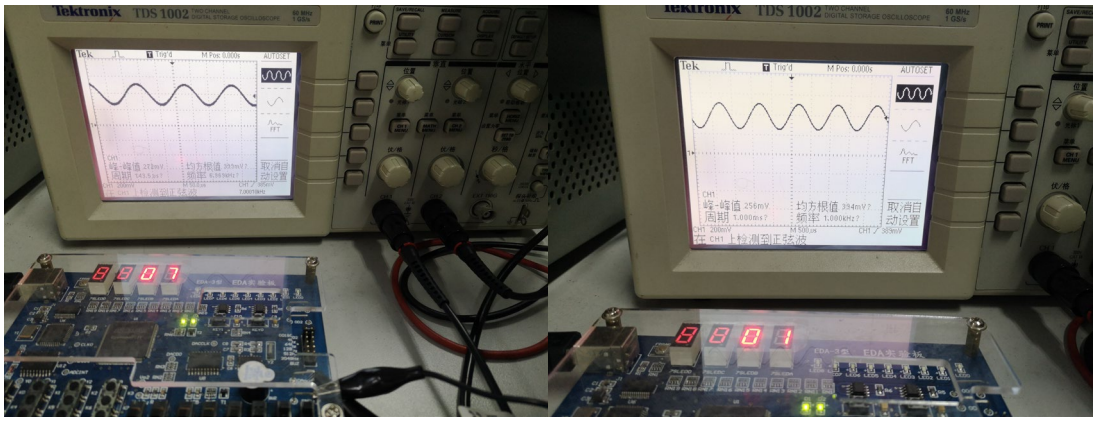
Node Name	Direction	Location	I/O Bank
CLK	Input	PIN_24	2
DAC[7]	Output	PIN_85	5
DAC[6]	Output	PIN_84	5
DAC[5]	Output	PIN_83	5
DAC[4]	Output	PIN_80	5
DAC[3]	Output	PIN_77	5
DAC[2]	Output	PIN_76	5
DAC[1]	Output	PIN_75	5
DAC[0]	Output	PIN_74	5
DACLK	Output	PIN_86	5
KEY0	Input	PIN_90	6
LEDA[6]	Output	PIN_124	7
LEDA[5]	Output	PIN_121	7
LEDA[4]	Output	PIN_120	7
LEDA[3]	Output	PIN_119	7
LEDA[2]	Output	PIN_115	7
LEDA[1]	Output	PIN_114	7
LEDA[0]	Output	PIN_113	7
LEDB[6]	Output	PIN_133	8
LEDB[5]	Output	PIN_132	8
LEDB[4]	Output	PIN_129	8
LEDB[3]	Output	PIN_128	8
LEDB[2]	Output	PIN_127	7
LEDB[1]	Output	PIN_126	7
LEDB[0]	Output	PIN_125	7
SW0	Input	PIN_73	5
SW1	Input	PIN_72	4
<<new node>>			

（3）编程下载

在编程窗口中单机左上方的“Hardware Setup”按钮，在打开的窗口中选择“USB-Blaster”。在编程模式“Mode”中选择“JTAG”，并用鼠标选中 Program/Configure 下方的小方框。单机“Start”按钮，即开始下载操作，当“Progress”显示 100%时，下载结束

6.测量结果


Tek
Trig'd
M Pos: 0.000s
AUTOSET
FFT
CH2
峰-峰值 2.88V
周期 12.84μs?
均方根值 3.93V?
频率 77.88kHz?
取消自动设置
CH2 2.00V
M 5.00μs
CH2 3.76V
78.1260kHz
Measure 2
信源
CH2
类型
频率
值
19.08kHz?
返回
M 25.0μs
CH2 3.80V
19.5315kHz
Tektronix
TDS 1002



五、实验小结

实验中遇到什么问题？如何解决？实验收获和建议。

编译时未选择将该 VHDL 或者.bdf 文件 Set as Top-Level Entity，导致实际载编译的不是想要编译的文件。

管脚锁定时同样也未 Set as Top-Level Entity，导致显示出的管脚错乱，有很多底层模块的管脚。

Quartus Ⅱ 15.0 版本中可能未安装 USB-Blaster 驱动，但其实在 Quartus 的安装目录下有驱动包，只需要进入电脑的设备管理器，更新驱动即可，驱动包地址选择 Quartus 安装根目录/altera\15.0\quartus\drivers 即可正常下载到 EDA 板。

更新底层模块的 VHDL 代码后，重新编译，选择 Create Symbol Files For Current File，即可更新，不需要重新替换顶层原理图的元件。