

# 基于可编程逻辑器件实现任意 波形发生器

摘要:

本文介绍了一种基于 EDA（电子设计自动化）技术及可编程逻辑器件设计的任意波形发生器，运用直接数字频率合成技术（DDS 技术）的基本工作原理，通过 Quartus II 5.1 软件和 VHDL 语言编程，对硬件结构和工作方式进行重构，再由可编程逻辑器件控制数据的输出，经 D/A 转换器转换成相应模拟信号。在设计过程中，以 Altera 公司的 fpga 芯片为设计器件。讨论了 DDS 技术的基本组成结构、工作原理和特点

关键词:

电子设计自动化 可编程逻辑器件 任意波形发生器 直接数字频率合成技术（DDS 技术）

# 目 录

引言.....	4
<b>1 方案的选择与比较.....</b>	<b>6</b>
1.1.方案比较.....	6
1.2.方案的选择.....	6
1.3.实现 DDS 的三种技术方法.....	7
1.4.总体设计思路.....	8
<b>2 具体系统设计.....</b>	<b>8</b>
2.1 正弦波的设计.....	9
2.2 三角波的设计.....	13
2.4 锯齿波的设计.....	15
2.5 方波的设计.....	16
2.5 波形选择与控制的设计.....	17
<b>3 设计结果与理论设计的比较.....</b>	<b>20</b>
<b>4 结束语.....</b>	<b>20</b>
致谢	

# 引言

通过这一段的学习，我完成了从对 FPGA 不懂到利用到实际项目中去过渡，中电网的培训使我更好的驾驭这种工具，在此感谢各位老师.通过这一段的学习我总结 FPGA/CPLD 在产品应用中有不可比拟的优势:

- 1 快速设计开发, FPGA/CPLD 极大的灵活行,使其在缩短设计时间上有很大的优势,对一个新的项目有时可以不需要对硬件进行任何修改,这是利用单片机单独实现的系统上难以完成的;
- 2 设计弹性: 由于 FPGA/CPLD 的规模和容量不断的提高,从而提高了设计系统时的弹性,可以根据用户的实际情况和要求做相应的修改;设计资源丰富,随着 FPGA/CPLD 行业的发展,其设计资源也随之剧增,而且可以很快找到已实现的系统的简要说明;
- 3 提供良好升级性能。FPGA/CPLD 具有较强的软件升级功能，利用业界标准的 HDL 和 FPGA/CPLD 制造商提供的基于平台和工具集方法的工具，可以很方便地实现软件在各代 FPGA/CPLD 中的无缝移植。在硬件升级方面，FPGA/CPLD 顾名思义就是现场可编程，因而能轻松升级，很好地满足需求变化，延长了产品寿命，有助于网络设备跟踪标准和协议的持续变化。
- 4 提供高抗扰能力。FPGA/CPLD 的串行连接技术可以减少引脚数量、减小接头尺寸、降低电磁干扰辐射（EMI）、提高信号完整性和更好地抵抗噪声，从而大大提升了系统的抗干扰能力，非常适用于对电磁辐射安全有特殊要求的应用环境。

综上所述，EDA 技术和 FPGA/CPLD 的发展，不仅可使系统的体积小、重量轻且功耗低，更重要的是可使系统的可靠性大大提高。另外,伴随着微控制器技术(mcu\dsp)的发展,我个人认为,两种技术必将走向融合的道路,这种趋势在 ATME1 公司推出的 FPSILC 芯片已经体现.

FPGA/CPLD 普及的另一个重要原因是 IP（知识产权）越来越被高度重视，

带有 IP 内核的功能模块在 ASIC 设计平台上的应用日益广泛。越来越多的设计人员，采用设计重要，将系统设计模块化，为设计带来了快捷和方便。并可以使每个设计人员充分利用软件代码。提高开发效率，减少上市时间，降低研发费用，缩短研发周期，降低风险。这是研究 EDA 技术必须关注的问题。在电子设计技术领域，可编程逻辑器件的广泛应用，为数字系统的设计带来极大的灵活性。高集成度，高速和高可靠是 CPLD/FPGA 最明显的特点。由于 CPLD/FPGA 的集成规模非常大，可利用先进的 EDA 工具进行电子系统设计和产品开发，在超高速战速决应用领域和实时测控方面有非常广阔的应用前景。

本论文就利用 FPGA/CPLD 器件实现 DDS（直接数字频率合成）技术,来验证 FPGA/CPLD 的优越性.

DDS（直接数字频率合成）技术是一种先进的频率合成技术，有着易于控制、相位连续、输出频率为宽频度高、分辨率高等特点。DDS 完全不同于我们已经熟悉的直接频率合成技术和锁相环频率合成技术，他的基本原理是采样定理，通过查表法产生波形，并且随着大规模集成电路技术的飞速发展，其优越性逐步显现出来

在现代电子测量仪器中，任意波形发生器（Arbitrary Waveform Generator，简称 AWG）作为最新一类的信号源，正日益受到人们的重视，国内外的许多科研单位和高等院校也纷纷着手研制 AWG。但目前使用的 AWG 大部分是利用分立元件实现的，体积大、可靠性差、准确度低。本文是一种采用基于 FPGA/CPLD 实现的 AWG 设计方法，用一片 FPGA/CPLD 完成几十片 MSI（Middle Scale Integration,中规模集成）电路才能完成的任务，并且波形合成采用了 DDS（Direct Digital Synthesis）技术。在设计过程中，通过运用 Quartus II 5.1 软件和 VHDL 编程语言，可以进行软件模拟检测设计的正确性，大大简化了系统结构，降低了成本，提高了系统的性能和可靠性。

# 1 方案的选择与比较

## 1.1 方案比较

要实现任意波形发生器有以下方案

a 方案一：采用单片机函数发生器（如 8038），8038 可同时产生正弦波、脉冲波，方法简单易行；用 D/A 转换器的输出来改变调节电压，也可以实现数控调整频

率，但步长难以满足要求，且频率稳定度不太高。

b 方案二：采用锁相式频率合成器，利用锁相环，将压控振荡器（VCO）的输出频率锁定在所需频率上，该方案性能良好，但难以达到输出频率覆盖系数的要求，且电路复杂，不适于产生低频信号。

c 方案三：采用直接数字频率合成器（DDS），可用硬件或软件实现。即用累加器按频率要求相对应的相位增量进行累加，再以累加相位值作为地址码，取存放于 ROM 中的波形数据，经 D/A 转换、滤波即得所需波形。方法简单，频率稳定度高，易于程控。

d 方案四：采用单片机控制动态生成程序。该方法引入动态编程和吞时钟技术，使用 8031 便可产生 50kHz 的正弦波。

## 1.2 方案的选择

DDS 问世之初，构成 DDS 元器件的速度的限制和数字化引起的噪音这两个主要缺点阻碍了 DDS 的发展与实际应用。近几年超高数字电路的发展以及对 DDS 的深入研究，DDS 的最高工作频率以及噪声性能已接近并达到锁相频率合成器相当的水平。随着这种频率合成技术的发展，现已广泛应用于通讯、导航、雷达、遥控遥测、电子对抗以及现代化的仪器仪表工业等领域。他有着易于控制、相位连续、输出频率稳定度高、分辨率高等优点。

DDS 技术的实现依赖于高速、高性能的数字器件。可编程逻辑器件以其速度高、规模大、可编程、以及有强大 EDA 软件支持等特性，十分适合实现 DDS 技术。

鉴于 DDS 目前的飞速发展和显著的优点及可编程逻辑器件十分适合实现 DDS 技术，我们自然想到采用 DDS 来实现该系统。

## 1.3.实现 DDS 的三种技术方法

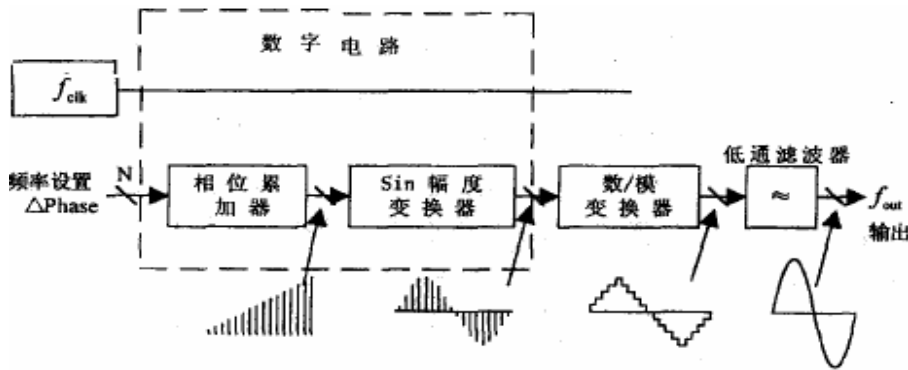
### 1.DDS 的基本原理

如图 所示，将要产生的波形数据存入波形存储器中，然后在参考脉冲的作用下，对输入的频率数据进行累加，并将累加器输出的一部分作为读取波形存储器的地址，将读出的波形数据经 D/A 转换为相应的电压信号，D/A 转换器输出的一系列的阶梯电压信号经低通滤波器滤波后便输出了光滑的合成波形的信号。DDS 的输出信号的频率为：

$$f_{out} = \frac{\Delta\text{Phase}}{2^N} f_{clk} \quad (1)$$

$$\Delta\text{Phase} = 2\pi \left( \frac{f_{out}}{f_{clk}} \right) \text{ 弧度} \quad (2)$$

其中:  $f_{out}$  为信号合成频率;  $f_{clk}$  为参考时钟频率;  $\Delta\text{Phase}$  为频率设置数据, 也称为频率控制字  $N$  为相位累加器的位数。



DDS 原理框图

DDS 的频率分辨率即最低频率为:  $\Delta f_{out} = \frac{1}{2^N} f_{clk} \quad (3)$

所以只要  $N$  足够大, DDS 可以得到很小的频率间隔。要改变 DDS 的输出信号的频率, 只要改变  $\Delta\text{Phase}$  即可。根据奈奎斯特抽样定理, DDS 的最大频率为:

$$f_{max} = \frac{1}{2} f_{clk}$$

## 2. 实现 DDS 有三种技术方法

### a 采用高性能 DDS 单片电路

随着微电子技术的飞速发展, 目前高超性能优良的 DDS 产品不断推出, 主要有 Qualcomm、AD、Sciteg 和 Stanford 等公司单片电路 (monolithic)。Qualcomm 公司推出 DDS 系列 Q2220、Q2230、Q2234、Q2240、Q2368, 其中 Q2368 的时钟频率为 130MHz, 分辨率为 0.03Hz, 变频时间为 0.1us; 美国 AD 公司推出了他们的 DDS 系列: AD9850、AD9851、可以实现线形调频的 AD9852、两路正交输出的 AD9854 以及以 DDS 为核心的 QPSK 调制器 AD9853、数字上变频器 AD9856 和 AD9857。AD 公司的 DDA 系列产品以其较高的性能价格比, 目前取得了极为广泛的应用。

### b 采用低频正弦波 DDS 单片电路

此方案的典型电路有 Micro Linear 公司的电源管理事业部推出低频正弦波 DDS 单片电路 ML2035 以其价格低廉、使用简单得到广泛应用。ML2035 特性:

(1) 输出低频为直流到 25kHz，在时钟输入为 12.352MHz 以外频率分辨率可达到 1.5Hz (-0.75~+0.75Hz)，输出正弦波信号的峰—峰值为  $V_{CC}$ ；(2) 高度集成比，无需或仅需极少的外接元件支持，自带 3~12MHz 晶体振荡电路；(3) 兼容的 3 线 SPI 串行输入口，带双缓冲，能方便地配合单片机使用；(4) 增益误差和总谐波失真很低。

ML2035 生成的频率较低 (0~25KHz)，一般应用于一些需要产生的频率为工频和音频的场合。如用 2 片 ML2035 产生多频互控信号，并与 AMS3104 (多频接收芯片) 或 ML2031/2032 (音频检波器) 配合，制作通信系统中的收发电路等。可编程正弦波发生器芯片 ML2035 设计巧妙，具有可编程、使用方便、价格低廉等优点，应用范围广泛。很适合需要低成本、高可靠性的低频正弦波信号的场合。

#### c 设计基于 FPGA/CPLD 的个人方法

DDS 技术的实现依赖于高速、高性能的数字器件。可编程逻辑器件以其速度快、规模大、可编程、以及有强大 EDA 软件支持等特性，十分适合实现 DDS 技术。目前 PLD 器件 (包括 CPLD、FPGA) 的生成厂商主要有 Altera, Xilinx 以及 Lattice 等。Altera 是一著名的 PLD 生成厂商，多年来一直占据着行业领先的地位。Altera 的 PLD 具有高性能、高集成度和高性价比的优点，此外它还提供了功能全面的开发工具和丰富的 IP 核、宏功能外它还提供了功能全面的开发工具和丰富的 IP 核、宏功能库等，因此 Altera 的产品获得了广泛的应用。虽然有的专用 DDS 芯片的功能也比较多，但控制方式却是固定的，因此不一定是我们所需要的。而利用 FPGA/CPLD 则可以需要方便地实现各种比较复杂的调频、调相和调幅功能，具有良好的实用性。就合成信号质量而言，专用 DDS 芯片由于采用特定的集成工艺，内部数字信号抖动很小，可以输出高质量的模拟信号；利用 FPGA/CPLD 也能输出高质量的信号，虽然达不到专业 DDS 芯片的水平，但信号精度误差也在允许范围之内。

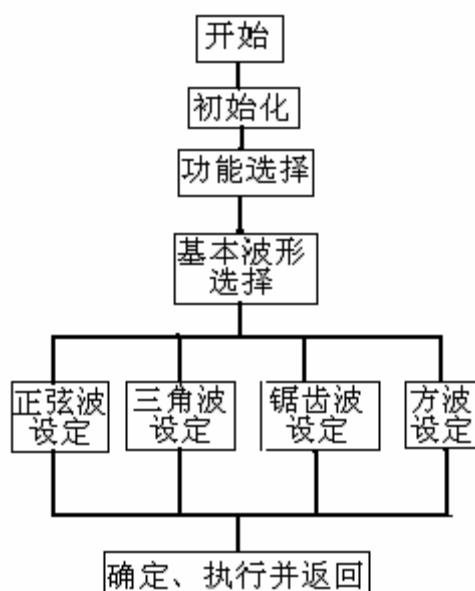
### 1.4 总体设计思路

将上述 DDS 的三种实现方案，采用集成 DDS 芯片固然能实现，但灵活性不强，故采用基于 CPLD 芯片的自行设计方案。采用自顶向下细化的设计方法，首先对波形生成子系统进行分析与方案选择入手。确认了波形产生子系统的实现方

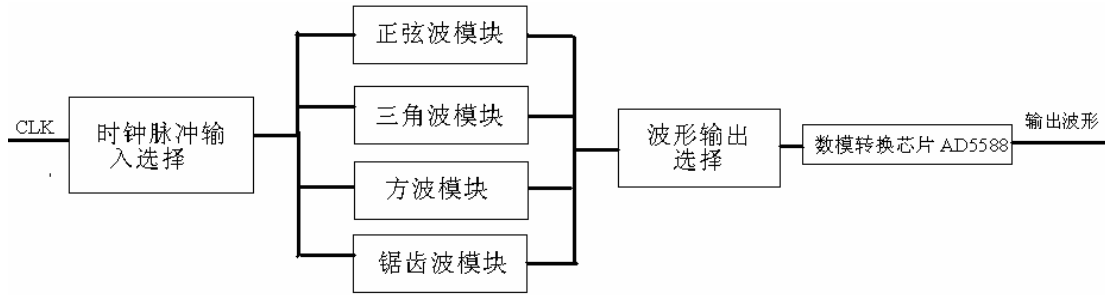
案后，再来考虑各参数控制子系统的实现。但是，基于使用最少的元件实现最好功能的想法，并不是完全依照方案三，在这过程中，我们选择不使用 ROM 芯片，而是使用 Quartus II 5.1 可编程逻辑器件开发软件和 VHDL 语言，采用个人算法对所实现的波形采样和编程，将程序进行软模拟和封装并下载到 FPGA/CPLD 中，由可编程逻辑器件控制数据的输出，再经过 D/A 转换输出所需要的模拟波形。

## 2 具体系统设计

本方案软件采用了结构化系统设计与结构化程序设计的方法，整个软件由顶向下分层分块，每个模块完成一项功能，并遵守上层模块调用下层模块，同层模块不能相互调用的原则，软件的模块如下图所示。



基于所做的是任意波形发生器，所以在波形选择上以四种基础波形为设计与实现的对象，即正弦波、三角波、锯齿波、方波。而四个波形作为同一个任意波形发生器里的四个部分，是有着同一个输入与输出，因此在设计上还需要对波形进行选择与控制的部分，通过对时钟脉冲输入的选择，使得四个波形模块只有一个输入为时钟脉冲，其他三个模块则输入始终为 0。在波形输出时，设计一个模块控制输出的波形是所要求输出的波形，在时钟脉冲选择与输出波形选择两模块之间，有个别情况需要注意，后面将有介绍，基本框图和如下：

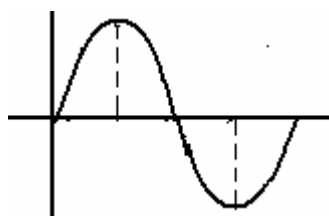


这是一个通过 VHDL 语言对信号发生器的电路进行硬件描述，把描述的整个电路通过动态配置写进芯片。输出的波形信号，要在示波器上观察其波形是否正确，就必须是模拟信号，所以要利用数/模芯片把其数字信号转换为模拟信号。

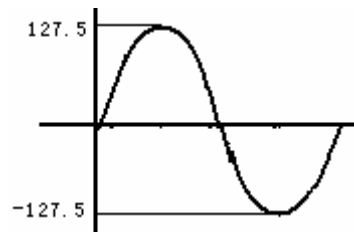
以下是对各个波形及波形的选择与控制的具体设计思路与步骤

### 2.1 正弦波的设计

由于正弦波的非线性，所以对算法有较严的要求。对于一个正弦波（如图 1），



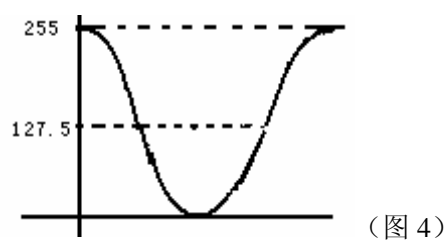
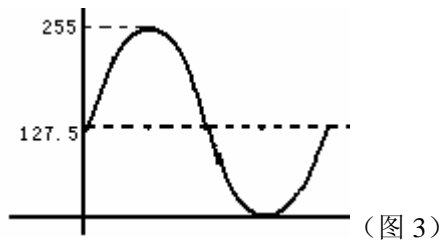
(图 1)



(图 2)

他在一个周期里变化有规律可循，由图和公式  $\sin a = \sin(\pi - a)$  可以看出，只需要算出 1/4 个周期的幅值，则可得出其他 3/4 个周期的幅值，而在我们所设计的过程中，也正是采用了这个规律来简化采样点数。由于 D/A 转换接口为 8 位输入，所以必须要考虑到采样个数和采样值的大小。如果仅以原始波形为基础进行采样，必然会出现小数和负数，这将使得二进制的转化和编程变的很困难。综合以上因素，首先要解决的问题就是消除小数和负数，使采样数值变成正整数，于是我们采用了以下的算法：

首先，将原始波形幅度扩大为原来的 127.5 倍，如（图 2）。然后将波形向上移动 127.5，如（图 3）。这时进行采样  $\sin a$ ，在简便又不影响波形输出且是误差允许范围内的情况下，小数点后的数值可以四舍五入，只取整数。为了简化采样与编程的需要，对（图 3）进行了变化，将正弦波左移 1/4 个周期，如（图 4），按余弦来采样，即  $\cos a$ 。在（0 到  $\pi$ ）之间取了 64 个  $\cos a$  点，而（ $\pi$  到  $2\pi$ ）之间的点则是（0 到  $\pi$ ）之间的点的反方向排列。采样点的计算公式为  $(\cos a + 1) * 127.5$



以下为正弦波的 VHDL 语言编程源程序

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity sin4 is
port(clk4:in std_logic;
      dd4:out integer range 255 downto 0);——输出范围为 0 到 255
end;
architecture dacc of sin4 is
  signal q: integer range 63 downto 0;
  begin
  process(clk4)
  begin
    if (clk4'event and clk4='1') then
      q<=q+1;
    end if;
  end process;
  process(q)
  begin
    case q is
      when 00=>dd4<=255;
      when 01=>dd4<=254;
      when 02=>dd4<=253;
      when 03=>dd4<=250;
      when 04=>dd4<=245;

```

when 05=>dd4<=240;  
when 06=>dd4<=234;  
when 07=>dd4<=226;  
when 08=>dd4<=218;  
when 09=>dd4<=208;  
when 10=>dd4<=198;  
when 11=>dd4<=188;  
when 12=>dd4<=176;  
when 13=>dd4<=165;  
when 14=>dd4<=152;  
when 15=>dd4<=140;  
when 16=>dd4<=128;  
when 17=>dd4<=115;  
when 18=>dd4<=103;  
when 19=>dd4<=90;  
when 20=>dd4<=79;  
when 21=>dd4<=67;  
when 22=>dd4<=57;  
when 23=>dd4<=47;  
when 24=>dd4<=37;  
when 25=>dd4<=29;  
when 26=>dd4<=21;  
when 27=>dd4<=15;  
when 28=>dd4<=10;  
when 29=>dd4<=5;  
when 30=>dd4<=2;  
when 31=>dd4<=1;  
when 32=>dd4<=0;  
when 33=>dd4<=1;  
when 34=>dd4<=2;

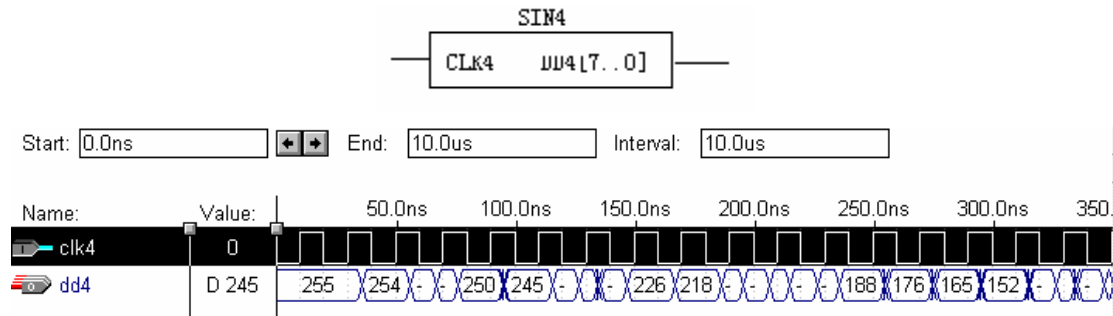
when 35=>dd4<=5;  
when 36=>dd4<=10;  
when 37=>dd4<=15;  
when 38=>dd4<=21;  
when 39=>dd4<=29;  
when 40=>dd4<=37;  
when 41=>dd4<=47;  
when 42=>dd4<=57;  
when 43=>dd4<=67;  
when 44=>dd4<=79;  
when 45=>dd4<=90;  
when 46=>dd4<=103;  
when 47=>dd4<=115;  
when 48=>dd4<=128;  
when 49=>dd4<=140;  
when 50=>dd4<=165;  
when 51=>dd4<=176;  
when 52=>dd4<=188;  
when 53=>dd4<=198;  
when 54=>dd4<=208;  
when 55=>dd4<=218;  
when 56=>dd4<=226;  
when 57=>dd4<=234;  
when 58=>dd4<=240;  
when 59=>dd4<=245;  
when 60=>dd4<=250;  
when 61=>dd4<=253;  
when 62=>dd4<=254;  
when 63=>dd4<=255;  
when others=>null;

```

end case;
end process;
end architecture;

```

在 Quartus II 5.1 下得到的电路模块图形和软件仿真数据：



## 2.2 三角波的设计

由于三角是由线形增加和线形递减的两个直线所构成，所以可以直接使用 VHDL 语言编程来实现三角波的采样和自加，当线形自加到最高点时，由控制语句控制其自减，直到减到最低点既 0 点时再继续重复以前过程，从而实现三角波数据的输出。以下是三角波 VHDL 语言编程源程序

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity sanj is
port(clk3:in std_logic;
      dd3:out std_logic_vector(7 downto 0));
end sanj;
architecture art of sanj is
    signal b:std_logic;
    signal c:std_logic_vector(7 downto 0);
begin
process(clk3)
begin
if (clk3'event and clk3='1') then
    if(b='0') then

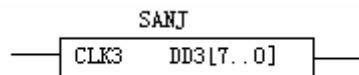
```

```

c<=c+1;
if(c=250) then
b<='1';
end if;
elsif(b='1') then
c<=c-1;
if(c=1) then
b<='0';
end if;
end if;
end if;
dd3<=c;
end if;
end process;
end art;

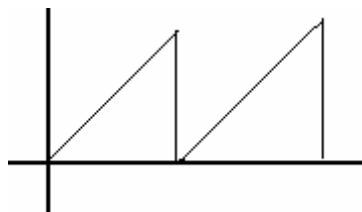
```

在 Quartus II 5.1 下得到的电路模块图形如下图：

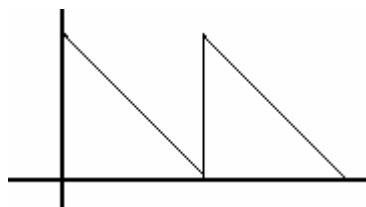


### 2.3 锯齿波的设计

对于锯齿波的采样，也可以完全用 VHDL 语言来实现采样和自加，当加到所要求的最高点时，由控制语句使其返回 0 点重复以前过程，从而实现锯齿波数据的输出。所要注意的是，锯齿波有上升锯齿波形和下降锯齿波形，如（图 5）（图 6）。所以必须设置一个控制按键，来控制波形，使达到所需要的上升波形或下降波形，程序中以 UP\_DOWN 来实现。



（图 5）



（图 6）

以下为锯齿波 VHDL 语言编程源程序

```

library ieee;
use ieee.std_logic.1164.all;

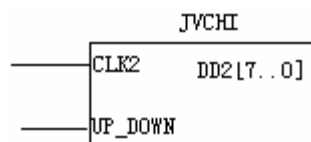
```

```

entity jvchi2 is
    potr(clk2,up_down: in std_logic;
          dd2:buffer integer range 255 downto 0);
end;
architecture one of jvchi2 is
    signal  d,temp:integer range 255 downto 0;
begin
    process(clk2)
    begin
        if(clk2'event and clk2='1')    then
            if temp<198 then    temp<=temp+2;
            else temp<=0;
            end if;
        end if;
    end process;
    process(temp,up_down)
    begin
        if up_down='0' then d<=temp;
    else d<=198-temp;
    end if;
    end process;
    dd2<=d;
end;

```

在 Quartus II 5.1 下得到的电路模块图形如下图：



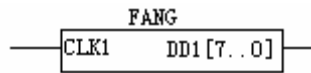
## 2.4 方波的设计

由于时钟脉冲输出即是方波波形，所以对方波的设计可以简化为直接输出时

钟脉冲信号。但是，在本设计中方波与其他三个波形要同步且要经过 D/A 转换，所以还需把时钟脉冲变成 8 位输出才可以经由 D/A 转换输出，具体过程可以由以下程序实现：

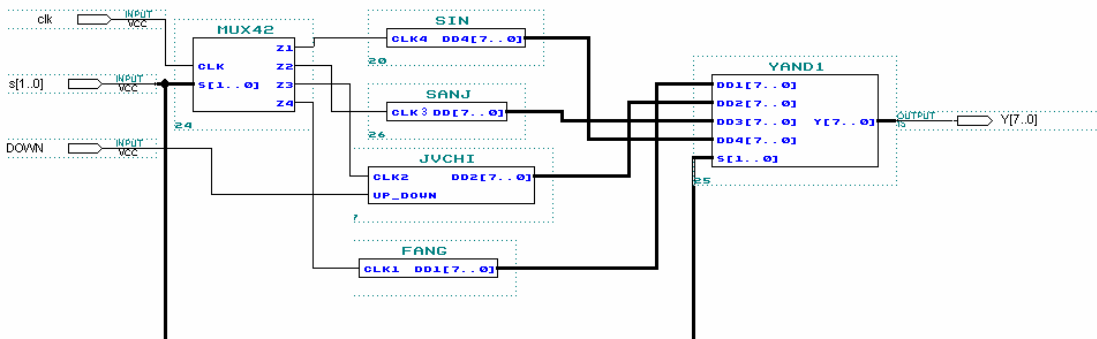
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity fang2 is
port(
clk1 : in std_logic;
dd1 : buffer integer range 255 downto 0);
end;
architecture dacc of fang2 is
signal q: integer range 1 downto 0;
begin
process(clk1)
begin
if (clk1'event and clk1='1') then
q<=q+1;
end if;
end process;
process(q)
begin
case q is
when 0=>dd1<=255;
when 1=>dd1<=0;
when others=>null;
end case;
end process;
end architecture;
```

在 Quartus II 5.1 下得到的电路模块图形如下图所示



## 2.5 波形选择与控制的设计

在前面已经分别设计出了四种基础波形，现在所需要做的是如何将四种波形融合在一起成为一个整体，并使他们能按操作人意愿输出所需要波形，这就需要波形选择与控制模块了。在我们的设计构思里，首先是让四个波形模块按意愿工作起来，使需要的波形模块能输出波形数据，其他的三个则的不输出波形数据。这个功能由时钟脉冲输入选择模块完成，在这个模块里，有两个控制开关，有 00、01、10、11 四个控制情况，这四种情况每一个针对一个波形模块——00 时正弦波模块，01 时三角波模块，10 时锯齿波模块，11 时方波模块。需要哪个波形工作时，按键开关输入哪种情况既可。当选择了一种情况时，所对应的一个波形模块输入的是时钟脉冲而能够工作输出波形数据，其他三个则始终输入为 0，所以不能输出波形。但是，不能输出波形并不说明不在工作，因为始终有个 0 信号输入，所以最后也能产生数据，是对波形有干扰的数据。因此，就需要输出波形选择模块来选择有用的波形，隔离干扰数据。为了达到时钟脉冲选择与输出波形选择的统一，同时也为了消除延迟，在输出波形选择模块里的两个控制开关是与时钟脉冲选择模块里的两个控制开关是同一个。这样当输入一种控制数据时，输出的波形也就是所需要的波形。将两个模块的控制开关设置成同一个控制开关，也减少了按键的数量，节省了资源降低了错误几率。下面是设计的连线总图及时钟脉冲选择与输出波形选择的程序：



1.时钟脉冲输入选择器 VHDL 语言编程源程序

```
library ieee;
use ieee.std_logic_1164.all;
```

```

entity mux42 is
    port(clk : in std_logic;
          s: in std_logic_vector(1 downto 0);
          z1,z2,z3,z4: out std_logic);
end entity mux42;
architecture art of mux42 is
begin
    process(s)
    begin
        case s is
            when "00"=>z1<=clk;
            when "01"=>z2<=clk;
            when "10"=>z3<=clk;
            when "11"=>z4<=clk;
            when others=>z1<=null;
                                z2<=null;
                                z3<=null;
                                z4<=null;
        end case;
    end process;
end art;

```

## 2.输出波形选择器 VHDL 语言编程源程序

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity yand1 is
    port(dd1,dd2,dd3,dd4 : in std_logic_vector (7 downto 0);
          s: in std_logic_vector(1 downto 0);
          y: : out std_logic_vector (7 downto 0));

```

```

        end entity yand1;
architecture art of yand1 is
    begin
        process(s)
        begin
        case s is
        when "00"=>y<=dd4;
        when "01"=>y<=dd3;
        when "10"=>y<=dd2;
        when "11"=>y<=dd1;
        when others=>null;
        end case;
        end process;
    end architecture art;

```

### 3 设计结果与理论设计的比较

经过构思，编程设计，模拟，下载到 FPGA/CPLD 中。软件模拟数据和示波器显示波形相吻合，仅在波形的稳定及光滑程度与理论波形有着一定差距，但这差距并不大，不影响设计结果、波形的观察和测量及该任意波形发生器的使用。造成这差距的原因主要有算法的选择，采样的点数及精确度，程序书写的精确与简练程度，以及输入时钟脉冲频率的选择。

### 4 结束语

由于时间比较短,我仅把最基本的部分做了,还有很多功能没有增加上去,比如:设置输出频率,调节占空比,调节幅度等功能,另外还可以利用 fpga 内部的 pll 提高输出频率,或者换种思路利用 DSP Builder 设计 DDS 等. 这些功能我会在以后逐步实现的,从而提高自己对 EDA 技术的掌握.这个项目虽然很小,但麻雀虽小,五脏俱全,对初学者不失为好的选择.

最后提一个建议:多讲些实例和软件的使用.

# 致谢

感谢信息产业部、中电网、Altera 公司能给予我这次设计的机会，让我认认真真的学到了一门技术，为我以后工作指明一个方向。