****

**2021-2022学年第2学期**

**数字系统设计工程实践**

**(课号: 103D47B01)**

**实 验 报 告**

|  |  |
| --- | --- |
| **实验名称：** | DDS波形发生器 |

|  |  |
| --- | --- |
| **学 院** | 信息科学与工程学院 |
| **班 级** | 阳明2班 |
| **姓 名** | 熊 康 |
| **学 号** | 206001232 |
| **指导教师** | 张 跃 军 |
| **完成时间** | 2022.03.24 |

目 录

[一、实验任务与要求 3](#_Toc100232479)

[1.1 实验目的 3](#_Toc100232480)

[1.2 实验设备与元器件 3](#_Toc100232481)

[1.3 实验内容 3](#_Toc100232482)

[二、系统原理与方案设计 3](#_Toc100232483)

[2.1 DDS原理 3](#_Toc100232484)

[2.1.1 工作过程 4](#_Toc100232485)

[2.1.2 改变输出信号的频率 4](#_Toc100232486)

[2.1.3 相位累加器 4](#_Toc100232487)

[2.1.4 正弦查询表 4](#_Toc100232488)

[2.1.5 DDS参数计算 4](#_Toc100232489)

[2.1.6 DDS的特点 5](#_Toc100232490)

[2.2 方案设计 5](#_Toc100232491)

[2.2.1 方案设计结构 5](#_Toc100232492)

[2.2.2 正弦查询表 6](#_Toc100232493)

[2.2.3 计算器 6](#_Toc100232494)

[2.2.4 信号频率控制器 7](#_Toc100232495)

[2.2.5 输出控制器 7](#_Toc100232496)

[2.2.6 方案参数的确定 7](#_Toc100232497)

[2.3 方案仿真 8](#_Toc100232498)

[三、系统实现与实验结果分析 8](#_Toc100232499)

[四、结论 9](#_Toc100232500)

[五、附录 10](#_Toc100232501)

# 实验任务与要求

## 1.1 实验目的

学习利用EDA技术和FPGA技术实现DDS波形发生器的设计。

## 1.2 实验设备与元器件

1. DSE-V数字电路实验平台 1台；
2. 计算机 1台；
3. Quartus II 软件 1套。

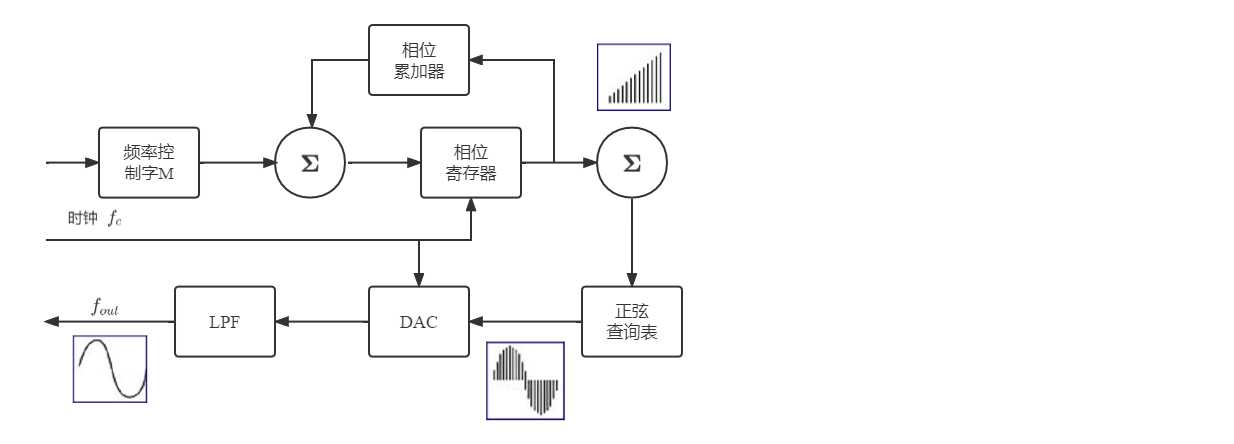
## 1.3 实验内容

**利用FPGA+DAC，设计一个DDS信号发生器。**

1. 分辨率优于10Hz；
2. ROM表长度10位、位宽8位；
3. 时钟频率：10MHz；
4. 输出最高频率时，每个输出周期不少于20个数据点；
5. 显示信号频率/频率控制字（可切换,十六进制显示）；
6. 直接输入频率控制字设置频率；
7. 用最少的N和M实现。

# 系统原理与方案设计

## 2.1 DDS原理



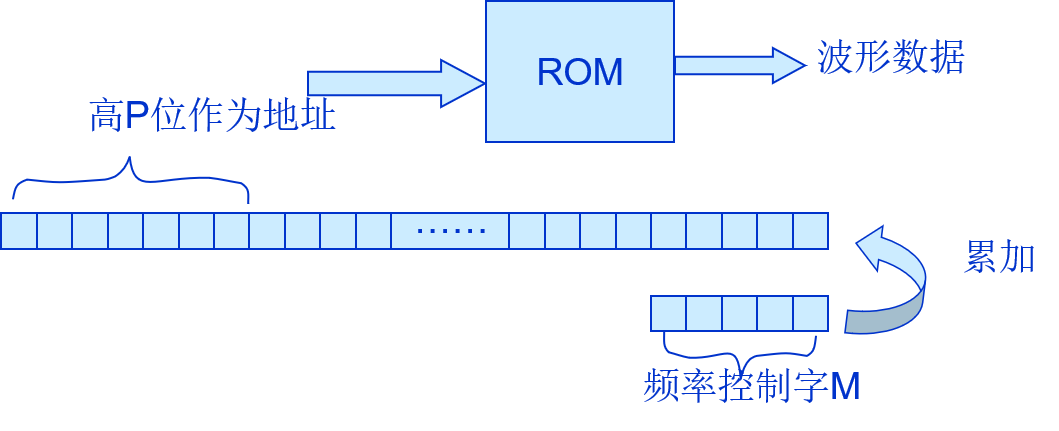
**图1 DDS基本结构框图**

2.1.1 工作过程

将存于数表中的数字波形,经D/A转换器和滤波，形成模拟量波形。

2.1.2 改变输出信号的频率

1. 改变查表寻址的时钟*频率*, 改变输出波形的频率；
2. 改变寻址的*步长，*改变输出信号的频率—DDS；
3. 步长即为对数字波形查表的相位增量.由累加器对相位增量进行累加，累加器的值作为查表地址；
4. 设相位累加器的位宽为N, Sin表的大小为2p，累加器的高P位用于寻址Sin表；
5. 时钟频率为*f*c, 若累加器按步进M累加直至溢出，称M为频率控制字。



**图2 信号频率控制原理图**

2.1.3 相位累加器

DDS系统的核心是相位累加器，它由一个加法器和一个相位寄存器组成；每来一个时钟，相位寄存器以步长增加，相位寄存器的输出与频率控制字(M)相加，然后输入到正弦查询表地址上。

2.1.4 正弦查询表

正弦查询表包含一个周期正弦波的数字幅度信息，每个地址对应正弦波中 0~360o 范围的一个相位点。查询表把输入的地址相位信息映射成正弦波幅度的数字量信号，驱动DAC，输出模拟量。

公式： （2.1）

2.1.5 DDS参数计算

相位寄存器每经过2N/M 个 *f*C 时钟后回到初始状态，相应地正弦查询表经过一个循环回到初始位置，整个DDS系统输出一个正弦波。

1. 输出正弦波频率：

（2.2）

1. M与输出*f*out和*f*C之间的关系:

（2.3）

1. **DDS的最小分辨率**通常用频率增量来表示频率合成器的分辨率。

（2.4）

这个增量也就是最低的合成频率。

1. **DDS的最大合成频率**最高的合成频率受奈奎斯特抽样定理的限制。

（2.5）

在实际运用中，为了保证信号的输出质量，输出频率不要高于时钟频率的33%,以避免混叠或谐波落入有用输出频带内。

2.1.6 DDS的特点

存在以下优点：

1. 超宽的相对宽带；
2. 超高的捷变速率（可实现跳频）；
3. 超细的分辨率；
4. 相位的连续性；
5. 输出波形灵活；
6. 可编程全数字化。

但存在杂散大的缺点。相位累加器相位舍位误差造成的杂散；幅度量化误差（由存储器有限字长引起）造成的杂散和DAC非理想特性造成的杂散。

频率上限目前还只能达到数百兆（主要是受DAC速度的限制）。

## 2.2 方案设计

2.2.1 方案设计结构

主体根据图1进行具体设计，此外为实现DAC，将DAC基准时钟与系统基准时钟信号clk相连。

|  |  |
| --- | --- |
| clk | 系统基准时钟信号输入 |
| rst | 系统重置信号输入 |
| clkm | 频率控制字脉冲信号输入 |
| car | 频率控制字位运算控制 |
| control | 数码管显示控制 |
| out | 数码管输出 |
| q | 数字波形输出 |

表1 输入输出量说明

|  |
| --- |
|  |

图3 方案设计结构图

2.2.2 正弦查询表

|  |  |
| --- | --- |
|  |  |
| 图4 功能模块正弦查询表 | 图5 .mif文件属性 |

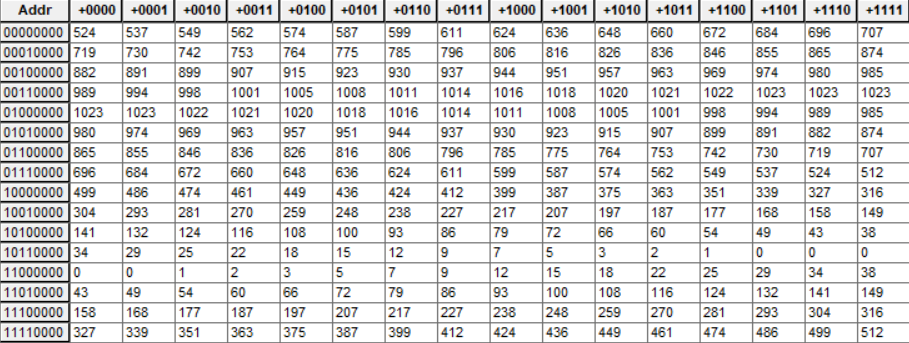


图6 正弦查询表.mif文件

通过公式（2.1），运用excel表格操作得到正弦数据后，新建.mif文件，导入数据，得到正弦查询表。

2.2.3 计算器

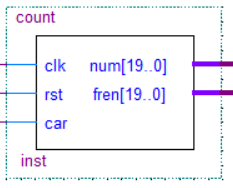


图7 功能模块计算器

通过寄存器和公式（2.2）运算实现计算频率控制字和输出频率。clk频率控制字脉冲输入，rst模块重置，car频率控制字进位控制，num显示频率控制字，fren显示输出频率。

2.2.4 信号频率控制器

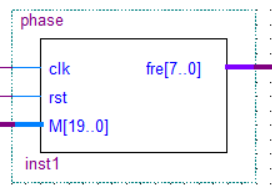


图8 功能模块信号频率控制器

根据原理2.1.2设计得到信号频率控制器。通过输入基准时钟频率clk和频率控制字M，得到当前输出频率fre。rst实现模块重置。

2.2.5 输出控制器

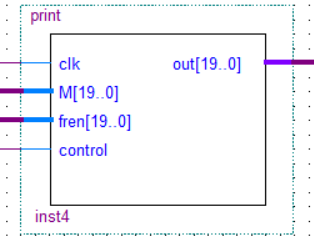


图9 功能模块输出控制器

通过control输出控制端，来实现对频率控制字M和输出频率fren的选择性输出。rst实现模块重置。clk为基准时钟信号。

2.2.6 方案参数的确定

**一、相位累加器的位宽N：**

在本次设计中，通过改变频率控制字M来实现改变输出频率。所以相位累加器的位宽N在设计时就被固定了。为了实现最小N和M，也为了实现分辨率优于10Hz，则根据式(2.2)得当时，使，即，所以。就是说N最小可取20。

**二、输出频率的显示：**

以Hz为单位，最低频率时M=1约9.5Hz，用一个数码管；最高频率高8位采样周期中每次采样应至少经过个周期，也就是说，M应为52428时，最高输出频率为500KHz，需要使用5个数码管，则在输出低频时不能显示小数位。

那么如果考虑在输出低频时，多用1个数码管显示小数位，就需要有一个进位标志。但在DSE-EP205目标板上，具有DAC模块的模式三和模式五只有5个数码管可供显示输出频率。而且LED灯不受系统控制，所以进位标志无法实现，即只能显示整数位。

## 2.3 方案仿真

|  |
| --- |
|  |

图10 方案仿真图

**分析：**由图可知，通过输入基准时钟信号clk，q输出正弦波。当clkm输入脉冲后，频率控制字M增加，输出频率增加，输出波形变密。当rst为0时，系统重置，符合要求。

# 系统实现与实验结果分析

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 图11 | 图12 | 图13 | 图14 |

**一、测试输出频率显示：**

1. 系统刚开始如图11所示，目标板选择模式5，数码管后5位（从右往左数5个）显示为0，按钮输入皆为0。

2. 图12所示，打开系统（最右侧按钮为1，rst=1），频率控制字加1（左侧第3个按钮按一次），数码管显示控制选择显示频率控制字（右侧第3个按钮为1，control=1），当前显示频率控制字M=1。

3. 图13所示，数码管显示控制选择显示输出频率（control=0），当前显示输出频率为9Hz。

4. 图14所示，实际输出频率为9.5Hz，在“方案参数的确定”中讨论了小数位的实现，所以只考虑显示整数位，符合要求。

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 图15 | 图16 | 图17 | 图18 |

5. 图15所示，当前显示频率控制字M=2。

6. 图16所示，当前显示输出频率为19Hz（数码管16进制，“13”表示16+3=19）。

7. 图17所示，实际输出频率为19.1Hz，相对误差为0.5%，符合要求。

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 图19 | 图20 | 图21 | 图22 |

8. 图18所示，当前显示频率控制字M=64。

9. 图19所示，当前显示输出频率为610Hz（十六进制262）。

10. 图20所示，实际输出频率为611.2Hz，相对误差为0.2%。 通过这三组数据可以发现，输出频率越大，数码管显示的频率误差越小。

**二、分析系统分辨率：**由图14可知，当N=1，M=1时，系统最小输出频率为9.5Hz，即分辨率为9.5Hz，优于19Hz。此时N为20，M为1。

**三、分析输出最高频率时，每个输出周期不少于20个数据点：**即高8位采样周期中每次采样应至少经过个周期，也就是说，M应为52428（二进制1100\_1100\_1100\_1100）（十六进制CCCC），而此时的最高频率为499992.37Hz≈500KHz，频率字调整如图21所示，输出图像如图22所示。此时N为20，M为52428。

# 四、结论

通过本次实验熟练掌握EDA技术和FPGA技术，并能实现DDS波形发生器的设计。

可以通过频率控制字改变输出频率，实现分频的目的。再通过查询正弦查询表，能够实现变化信号的不同频率地周期性输出。然后经过DAC处理后，从数字量转变成模拟信号。再通过增加计算和输出显示模块，还能增加各种显示输出功能。但唯一遗憾地是，在硬件限制的情况下，不能够实现小数位的显示。

# 五、附录

1. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
2. Author:  熊康
3. E-mail： 1390000666@qq.com
4. Device:  DSE-EP2C5
5. Tool:    Quartus 9.0
6. Function:（计算器）计算频率控制字和输出频率
7. Version: 2022-3-17 v1.0
8. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
9. module count(clk,rst,num,car,fren);
10. input clk,rst,car;
11. output reg [19:0] num;
12. output reg [19:0] fren;
14. always @(posedge clk or negedge rst) begin
15. //用于控制频率控制字
16. **if**(!rst)
17. num <= 0;
18. **else**
19. **if**(!car)
20. num <= num + 1; //M+1
21. **else**
22. num <= {num[18:0],1'b0}; //M\*2
23. end
25. always @(negedge clk or negedge rst) begin
26. **if**(!rst)
27. fren <= 0;
28. **else**
29. //计算输出频率
30. fren[19:0] <= (num \* (10000000 >> 18) >>2);
31. end
33. endmodule
34. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
35. Author:  熊康
36. E-mail： 1390000666@qq.com
37. Device:  DSE-EP2C5
38. Tool:    Quartus 9.0
39. Function:相位寄存器
40. Version: 2022-3-17 v1.0
41. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
42. module phase(clk,rst,M,fre);
43. input clk,rst;
44. input [19:0] M; //M：频率控制字
45. output reg [7:0] fre;  //输出的相位
46. reg [19:0] N;  //相位寄存
48. always @(posedge clk or negedge rst) begin
49. **if**(!rst)
50. N <= 0;
51. **else** begin
52. N <= N + M;
53. fre[7:0] <= N[19:12];
54. end
56. end
58. endmodule
59. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
60. Author:  熊康
61. E-mail： 1390000666@qq.com
62. Device:  DSE-EP2C5
63. Tool:    Quartus 9.0
64. Function:输出控制器(用于控制输出频率控制字和输出频率)
65. Version: 2022-3-24 v1.0
66. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
67. module print(clk,M,fren,control,out);
68. input [19:0] M;
69. input [19:0] fren;
70. input clk,control;
71. output reg [19:0] out;
73. always @(posedge clk) begin
74. **if**(control==1)
75. out[19:0] <= M[19:0];
76. **else**
77. out[19:0] <= fren[19:0];
78. end
80. endmodule
81. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
82. Author:  熊康
83. E-mail： 1390000666@qq.com
84. Device:  DSE-EP2C5
85. Tool:    Quartus 9.0
86. Function:正弦查询表
87. Version: 2022-3-24 v1.0
88. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
90. `timescale 1 ps / 1 ps
92. module sinphase (
93. address,
94. clock,
95. q);
97. input   [7:0]  address;
98. input     clock;
99. output  [9:0]  q;
100. `ifndef ALTERA\_RESERVED\_QIS
102. `endif
103. tri1      clock;
104. `ifndef ALTERA\_RESERVED\_QIS
106. `endif
108. wire [9:0] sub\_wire0;
109. wire [9:0] q = sub\_wire0[9:0];
111. altsyncram  altsyncram\_component (
112. .clock0 (clock),
113. .address\_a (address),
114. .q\_a (sub\_wire0),
115. .aclr0 (1'b0),
116. .aclr1 (1'b0),
117. .address\_b (1'b1),
118. .addressstall\_a (1'b0),
119. .addressstall\_b (1'b0),
120. .byteena\_a (1'b1),
121. .byteena\_b (1'b1),
122. .clock1 (1'b1),
123. .clocken0 (1'b1),
124. .clocken1 (1'b1),
125. .clocken2 (1'b1),
126. .clocken3 (1'b1),
127. .data\_a ({10{1'b1}}),
128. .data\_b (1'b1),
129. .eccstatus (),
130. .q\_b (),
131. .rden\_a (1'b1),
132. .rden\_b (1'b1),
133. .wren\_a (1'b0),
134. .wren\_b (1'b0));
135. defparam
136. altsyncram\_component.clock\_enable\_input\_a = "BYPASS",
137. altsyncram\_component.clock\_enable\_output\_a = "BYPASS",
138. altsyncram\_component.init\_file = "demo.mif",
139. altsyncram\_component.intended\_device\_family = "Cyclone II",
140. altsyncram\_component.lpm\_hint = "ENABLE\_RUNTIME\_MOD=NO",
141. altsyncram\_component.lpm\_type = "altsyncram",
142. altsyncram\_component.numwords\_a = 256,
143. altsyncram\_component.operation\_mode = "ROM",
144. altsyncram\_component.outdata\_aclr\_a = "NONE",
145. altsyncram\_component.outdata\_reg\_a = "CLOCK0",
146. altsyncram\_component.widthad\_a = 8,
147. altsyncram\_component.width\_a = 10,
148. altsyncram\_component.width\_byteena\_a = 1;

151. endmodule