

## 5.2 线性序列机与串行接口 DAC 驱动设计与验证

### 本章导读

数模转换器即 D/A 转换器，或简称 DAC (Digital to Analog Converter)，是指将数字信号转变为模拟信号电子元件。

DAC 的内部电路构成无太大差异，一般按输出是电流还是电压、能否作乘法运算等进行分类。大多数 DAC 由电阻阵列和  $n$  个电流开关(或电压开关)构成，按数字输入值切换开关，产生比例于输入的电流(或电压)。此外，也有为了改善精度而把恒流源放入器件内部的。DAC 可分为电压型和电流型两大类，电压型 DAC 有权电阻网络、T 型电阻网络和树形开关网络等；电流型 DAC 有权电流型电阻网络和倒 T 型电阻网络等。

电压输出型(如 TLV5618)。电压输出型 DAC 虽有直接从电阻阵列输出电压的，但一般采用内置输出放大器以低阻抗输出。直接输出电压的器件仅用于高阻抗负载，由于无输出放大器部分的延迟，故常作为高速 DAC 使用。

电流输出型(如 THS5661A)。电流输出型 DAC 很少直接利用电流输出，大多外接电流-电压转换电路得到电压输出，后者有两种方法：一是只在输出引脚上接负载电阻而进行电流-电压转换，二是外接运算放大器。

乘算型(如 AD7533)。DAC 中有使用恒定基准电压的，也有在基准电压输入上加交流信号的，后者由于能得到数字输入和基准电压输入相乘的结果而输出，因而称为乘算型 DAC。乘算型 DAC 一般不仅可以进行乘法运算，而且可以作为使输入信号数字化地衰减的衰减器及对输入信号进行调制的调制器使用。

一位 DAC。一位 DAC 与前述转换方式全然不同，它将数字值转换为脉冲宽度调制或频率调制的输出，然后用数字滤波器作平均化而得到一般的电压输出，用于音频输出等场合。

本章以 TLV5618 为例介绍 DAC 的工作原理及时序图解释，并用线性序列机 (LSM) 来描述时序图进而正确驱动此类设备。在 Quartus Prime 软件中，使用 ISSP 工具输入希望输出的电压值，控制 FPGA 进而操作 TLV5618 芯片输出对应的电压值。

### 5.2.1 DAC 芯片概述及电路设计

#### 1. TLV5618 型 DAC 内部工作原理

本章使用的 DAC 芯片为 TLV5618，其芯片内部结构如图 24.1 所示。TLV5618 是一个基于电压输出型的双通道 12 位单电源数模转换器，其由串行接口、一个速度和电源控制器、电阻网络、2 倍增益的输出缓冲器组成。

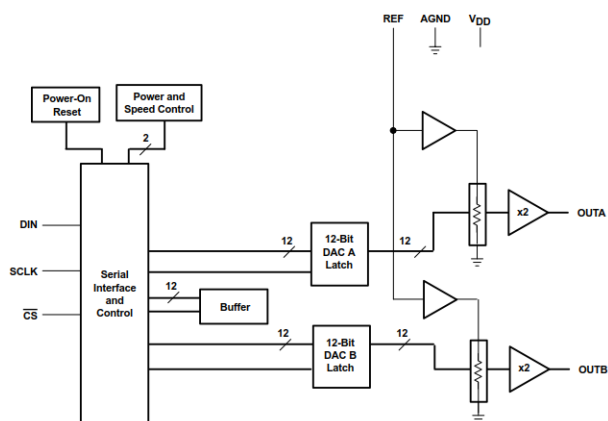


图 24.1 TLV5618 芯片内部框图

TLV5618 使用 CMOS 电平兼容的三线制串行总线与各种处理器进行连接，接收控制器发送的 16 位的控制字，这 16 位的控制字被分为 2 个部分，包括 4 位的编程位，12 位的数据位。

## 2. TLV5618 型 DAC 芯片引脚功能

TLV5618 芯片引脚及功能描述如表 24.1 所示。

表 24.1 芯片引脚功能描述

| 引脚名             | 编号 | I/O | 功能描述                  |
|-----------------|----|-----|-----------------------|
| DIN             | 1  | I   | 数字串行数据输入              |
| SCLK            | 2  | I   | 数字串行时钟输入              |
| $\overline{CS}$ | 3  | I   | 片选端。低电平数据输入有效，用作使能输入端 |
| OUTA            | 4  | O   | DAC A 模拟电压输出          |
| AGND            | 5  | 供电  | 地                     |
| REF             | 6  | I   | 模拟基准电压输入              |
| OUTB            | 7  | O   | DAC B 模拟电压输出          |
| VDD             | 8  | 供电  | 正电源                   |

## 3. TLV5618 电路设计

TLV5618 与 FPGA 采用三线制 SPI 通信，其电路图如图 24.2 所示。其中 TLV5618 的参考电压由 LM4040-2.0 提供，LM4040-2.0 是一个专用于 12 位精度场合的精密参考源，输出电压为 2.048V。

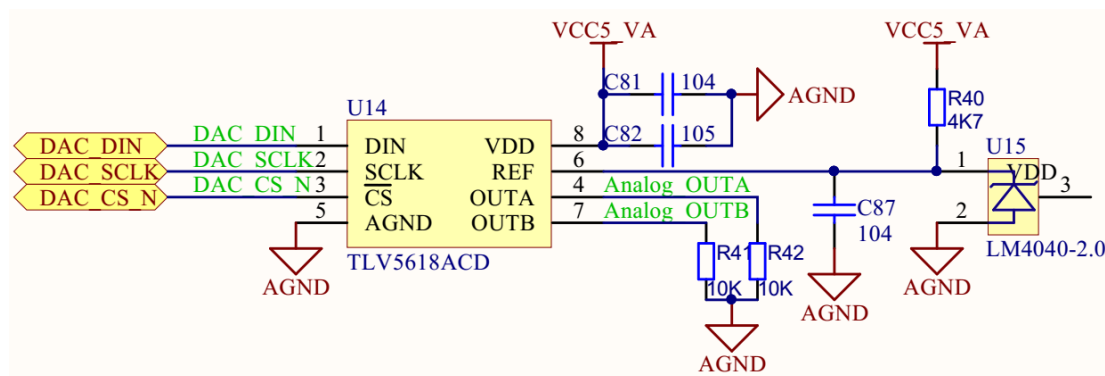


图 24.2 TLV5618 部分电路图

## 5.2.2 TLV5618 型 DAC 芯片输出电压计算原理

TLV5618 是由两个电阻网络来实现两路数模转换，每路 DAC 的核心是一个拥有 4096 ( $2^{12}$ ) 个节点的电阻，对应了 4096 种不同的组合，每个电阻网络的一段连接到 GND，另一端来自参考电压经过缓冲器后的输出。如果不考虑其他情况，该电阻网络型 DAC 的输出电压范围应该为  $0V \sim V_{REF}$ ，对应到 AC620 板上的电路，即  $0V \sim 2.048V$ 。另外在每个 DAC 通道的电阻网络电压输出后级，连接了一个 2 倍增益的轨对轨放大器。将电阻网络 DAC 单元的输出电压放大为 2 倍后输出到管脚。所以，TLV5618 芯片的实际输出电压范围为  $0V \sim 2 * V_{REF}$ ，对应到 AC620 板上的电路，即  $0V \sim 4.096V$ 。当芯片上电时，DAC 的值全部被复位到 0。每个 DAC 通道的输出可由下列公式计算得出：

$$V_o (\text{DAC A|B}) = 2 * \text{REF} * \text{CODE} / 2^{12} \quad (V)$$

其中，REF 是基准电压，本电路中为 2.048V；CODE 是数字电压输入值，范围 0 到  $2^{12}-1$ 。当串行控制字中的数据部分为 0~4095d 时，与输出电压对应关系如表 24.2 所示。需注意的是满量程输出电压由基准电压决定，可见下表。

表 24.2 输出电压与控制字对应关系

| D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | 输出电压                           |
|-----|-----|----|----|----|----|----|----|----|----|----|----|--------------------------------|
| 0   | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0V                             |
| 0   | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | $2 * (1/4096) * \text{REF}$    |
| .   | .   | .  | .  | .  | .  | .  | .  | .  | .  | .  | .  | .                              |
| .   | .   | .  | .  | .  | .  | .  | .  | .  | .  | .  | .  | .                              |
| 0   | 1   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | $2 * (1023/4096) * \text{REF}$ |
| 1   | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | $2 * (1024/4096) * \text{REF}$ |
| .   | .   | .  | .  | .  | .  | .  | .  | .  | .  | .  | .  | .                              |
| .   | .   | .  | .  | .  | .  | .  | .  | .  | .  | .  | .  | .                              |
| 1   | 1   | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | $2 * \text{REF}$               |

## 5.2.3 TLV5618 型 DAC 接口时序

当片选( $\overline{\text{CS}}$ )信号为低电平时，数据在每个 SCLK 信号的下降沿被移入芯片内部的寄存器。16 位的数据按照高位在前，低位在后的顺序依次移入。当 16 位的数据移入完毕后，在第 16 个 SCLK 信号的下降沿之后的一个 SCLK 信号上升沿，据根数据中的控制位，将数据制位

The timing diagram illustrates the relationship between the Serial Clock (SCLK), Data In (DIN), and Chip Select (CS) signals. The SCLK signal is a periodic square wave with periods labeled 1 through 6. The DIN signal is a data bus where data is captured on the rising edge of SCLK. The CS signal is an active-low pulse that must be held low for a duration of  $t_{su}(CS-CK)$  before the first SCLK rising edge and for  $t_{su}(C16-CS)$  after the last SCLK rising edge. Key timing parameters shown include  $t_{su}(D)$  (setup time for DIN),  $t_{h}(D)$  (hold time for DIN),  $t_{w}(L)$  (low pulse width for SCLK), and  $t_{w}(H)$  (high pulse width for SCLK).

TLV5618 的 16 位数据格式如下:

[illegible]

表 24.4 SPD、PWR 功能描述

|    |      |      |
|----|------|------|
|    | SPD  | PWR  |
| 数值 | 功能描述 |      |
| 1  | 快速模式 | 掉电模式 |
| 0  | 低速模式 | 正常工作 |

### 表 24.5 R1、R0 功能描述

| R1 | R0 | 描述                           |
|----|----|------------------------------|
| 0  | 0  | 写数据到 DAC B 和缓冲区              |
| 0  | 1  | 写数据到缓冲区                      |
| 1  | 0  | 写数据到 DAC A 同时把缓冲区内容更新到 DAC B |
| 1  | 1  | 保留                           |

1. 设置 DAC A 输出，选择快速模式：写新的 DAC A 的值，更新 DAC A 输出。DAC A 的输出在 D0 后的时钟上升沿更新。

|     |     |     |     |              |     |    |    |    |    |    |    |    |    |    |     |
|-----|-----|-----|-----|--------------|-----|----|----|----|----|----|----|----|----|----|-----|
| D15 | D14 | D13 | D12 | D11          | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0  |
| 1   | 1   | 0   | 0   | 新的 DAC A 输出值 |     |    |    |    |    |    |    |    |    |    | LSB |

[illegible]

a.写 DAC B 的数据到缓冲区:

|     |     |     |     |                    |     |    |    |    |    |    |    |    |    |    |    |
|-----|-----|-----|-----|--------------------|-----|----|----|----|----|----|----|----|----|----|----|
| D15 | D14 | D13 | D12 | D11                | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| 0   | 0   | 0   | 1   | MSB新的 DAC B 输出值LSB |     |    |    |    |    |    |    |    |    |    |    |

b.写新的 DAC A 的值并且同时更新 DAC A 和 B:

| D15 | D14 | D13 | D12 | D11          | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0  |
|-----|-----|-----|-----|--------------|-----|----|----|----|----|----|----|----|----|----|-----|
| 1   | 0   | 0   | 0   | 新的 DAC A 输出值 |     |    |    |    |    |    |    |    |    |    | LSB |

4. 设置掉电模式: ×=不关心

| D15 | D14 | D13 | D12 | D11         | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|-----|-----|-----|-----|-------------|-----|----|----|----|----|----|----|----|----|----|----|
| ×   | ×   | 1   | ×   | × × × × × × |     |    |    |    |    |    |    |    |    |    |    |

## 5.2.4 线性序列机设计思想与接口时序设计

从图 24.3 中可以看出,该接口的时序是一个很有规律的序列,SCLK 信号什么时候该由低变高,什么时候由高变低。DIN 信号什么时候该传输哪一位数据,都是可以根据时序参数唯一确定下来的。

这样就可以将该数据波形放到以时间为横轴的一个二维坐标系中,纵轴就是每个信号对应的状态:

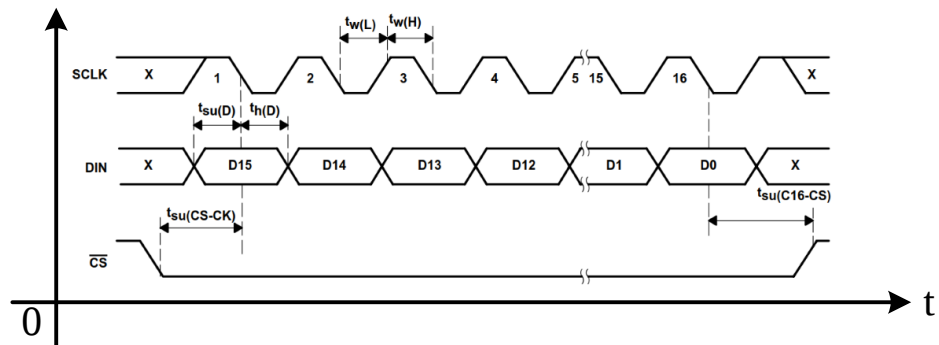


图 24.4 时序图数字化

因此只需要在逻辑中使用一个计数器来计数,然后每个计数值时就相当于在 t 轴上对应了一个相应的时间点,那么在这个时间点上,各个信号需要进行什么操作,直接赋值即可。

经查阅手册可知器件工作频率 SCLK 最大为 20MHz,设计时留下一定余量,因此这里定义其工作频率为 12.5MHz。设置一个两倍于 SCLK 的采样时钟 SCLK2X,使用 50M 系统时钟二分频而来即 SCLK2X 为 25MHz。针对 SCLK2X 进行计数来确定图 24.4 中各个信号的状态。可得出每个时间点对应信号操作详表。

表 24.6 时间点对应信号操作表

| 计数器值 | 对应信号状态                               |
|------|--------------------------------------|
| 0    | CS_N = 0;SCLK= 1;DIN = DAC_DATA[15]; |
| 1    | SCLK= 0;                             |
| 2    | SCLK= 1;DIN = DAC_DATA[14];          |
| 3    | SCLK= 0;                             |
| 4    | SCLK= 1;DIN = DAC_DATA[13];          |
| 5    | SCLK= 0;                             |
| 6    | SCLK= 1;DIN = DAC_DATA[12];          |
| 7    | SCLK= 0;                             |
| 8    | SCLK= 1;DIN = DAC_DATA[11];          |

|    |                             |
|----|-----------------------------|
| 9  | SCLK= 0;                    |
| 10 | SCLK= 1;DIN = DAC_DATA[10]; |
| 11 | SCLK= 0;                    |
| 12 | SCLK= 1;DIN = DAC_DATA[9];  |
| 13 | SCLK= 0;                    |
| 14 | SCLK= 1;DIN = DAC_DATA[8];  |
| 15 | SCLK= 0;                    |
| 16 | SCLK= 1;DIN = DAC_DATA[7];  |
| 17 | SCLK= 0;                    |
| 18 | SCLK= 1;DIN = DAC_DATA[6];  |
| 19 | SCLK= 0;                    |
| 20 | SCLK= 1;DIN = DAC_DATA[5];  |
| 21 | SCLK= 0;                    |
| 22 | SCLK= 1;DIN = DAC_DATA[4];  |
| 23 | SCLK= 0;                    |
| 24 | SCLK= 1;DIN = DAC_DATA[3];  |
| 25 | SCLK= 0;                    |
| 26 | SCLK= 1;DIN = DAC_DATA[2];  |
| 27 | SCLK= 0;                    |
| 28 | SCLK= 1;DIN = DAC_DATA[1];  |
| 29 | SCLK= 0;                    |
| 30 | SCLK= 1;DIN = DAC_DATA[0];  |
| 31 | SCLK= 0;                    |
| 32 | SCLK= 1;                    |
| 33 | CS_N = 1;                   |

线性序列机计数器的控制逻辑判断依据，如表 24.7 所示。

表 24.7 计数器功能判断条件

| 基本条件              | 对 SCLK_GEN_CNT 操作                     |
|-------------------|---------------------------------------|
| SCLK_GEN_CNT < 33 | SCLK_GEN_CNT<=<br>SCLK_GEN_CNT+ 1'b1; |
| SCLK_GEN_CNT = 33 | SCLK_GEN_CNT<= 0;                     |

以上就是通过线性序列机设计接口时序的一个典型案例，可以看到，线性序列机可以大大简化设计思路。线性序列机的设计思想就是使用一个计数器不断计数，由于每个计数值都会对应一个时间，那么当该时间符合需要操作信号的时刻时，就对该信号进行操作。这样，就能够轻松的设计出各种时序接口了。

### 5.2.5 基于线性序列机的 DAC 驱动设计

设计 TLV5618 接口逻辑的模块如图 24.8 所示。

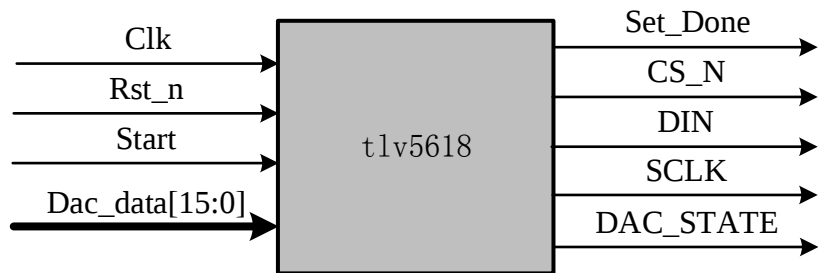


图 24.4 模块接口示意图

其中，每个端口的功能描述如表 24.6 所示。

表 24.6 模块端口功能描述

| 端口名称           | I/O | 端口功能描述                                    |
|----------------|-----|-------------------------------------------|
| Clk            | I   | 为控制器的工作时钟，频率为 50MHz，                      |
| Rst_n          | I   | 控制器复位，低电平复位                               |
| Start          | I   | 模块使能控制                                    |
| Dac_data[15:0] | I   | 控制器控制字                                    |
| Set_Done       | O   | 更新 DAC 完成标志，每次完成更新产生一个高电平脉冲，脉冲宽度为 1 个时钟周期 |
| CS_N           | O   | TLV5618 的 CS_N 接口                         |
| DIN            | O   | TLV5618 的 DIN 接口                          |
| SCLK           | O   | TLV5618 的 SCLK 接口                         |
| DAC_STATE      | O   | 模块状态标识，低电平时为忙标志，高电平为空闲状态                  |

生成使能信号，当输入使能信号有效后便将使能信号 en 置 1，当转换完成信号有效时便将其重新置 0。

```

reg en; //转换使能信号
always@(posedge Clk or negedge Rst_n)
if(!Rst_n)
    en <= 1'b0;
else if(Start)
    en <= 1'b1;
else if(trans_done)
    en <= 1'b0;
else
    en <= en;

```

在数据手册中 SCLK 的频率范围最大为 20MHz。这里为了方便适配不同的频率需求率，设置了一个可调的计数器，改变 DIV\_PARAM 的值即可改变 DAC 工作频率。根据表 24.6 中可以看出，需要根据计数器的值周期性的产生 SCLK 时钟信号，这里可以将计数器的值等倍数放大，形成过采样。这里产生一个两倍于 SCLK 的信号，命名为 SCLK2X。

首先编写分频计数器，时钟 SCLK2X 的计数器。

```

//生成 2 倍 SCLK 使能时钟计数器
reg [7:0]DIV_CNT; //分频计数器
always@(posedge Clk or negedge Rst_n)
if(!Rst_n)
    DIV_CNT <= 4'd0;

```

```

else if(en)begin
    if(DIV_CNT == (DIV_PARAM - 1'b1))
        DIV_CNT <= 4'd0;
    else
        DIV_CNT <= DIV_CNT + 1'b1;
end else
    DIV_CNT <= 4'd0;

```

根据使能信号以及计数器状态生成 SCLK2X 时钟。

```

//生成 2 倍 SCLK 使能时钟计数器
always@(posedge Clk or negedge Rst_n)
if(!Rst_n)
    SCLK2X <= 1'b0;
else if(en && (DIV_CNT == (DIV_PARAM - 1'b1)))
    SCLK2X <= 1'b1;
else
    SCLK2X <= 1'b0;

```

每当使能转换后，对 SCLK2X 时钟进行计数。

```

always@(posedge Clk or negedge Rst_n)
if(!Rst_n)
    SCLK_GEN_CNT <= 6'd0;
else if(SCLK2X && en)begin
    if(SCLK_GEN_CNT == 6'd33)
        SCLK_GEN_CNT <= 6'd0;
    else
        SCLK_GEN_CNT <= SCLK_GEN_CNT + 1'd1;
end else
    SCLK_GEN_CNT <= SCLK_GEN_CNT;

```

根据 SCLK2X 计数器的值来确认工作状态以及数据传输进程。

```

//依次将数据移出到 DAC 芯片
always@(posedge Clk or negedge Rst_n)
if(!Rst_n)begin
    DIN <= 1'b1;
    SCLK <= 1'b0;
    CS_N <= 1'b1;
end else if(!Set_Done && SCLK2X) begin
    case(SCLK_GEN_CNT)
        0:
            begin
                CS_N <= 1'b0;
                DIN <= r_DAC_DATA[15];
                SCLK <= 1'b1;

```

```

        end
    1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31:
    begin
        SCLK <= 1'b0;
    end

    2: begin DIN <= r_DAC_DATA[14]; SCLK <= 1'b1; end
    4: begin DIN <= r_DAC_DATA[13]; SCLK <= 1'b1; end
    6: begin DIN <= r_DAC_DATA[12]; SCLK <= 1'b1; end
    8: begin DIN <= r_DAC_DATA[11]; SCLK <= 1'b1; end
    10: begin DIN <= r_DAC_DATA[10]; SCLK <= 1'b1; end
    12: begin DIN <= r_DAC_DATA[9]; SCLK <= 1'b1; end
    14: begin DIN <= r_DAC_DATA[8]; SCLK <= 1'b1; end
    16: begin DIN <= r_DAC_DATA[7]; SCLK <= 1'b1; end
    18: begin DIN <= r_DAC_DATA[6]; SCLK <= 1'b1; end
    20: begin DIN <= r_DAC_DATA[5]; SCLK <= 1'b1; end
    22: begin DIN <= r_DAC_DATA[4]; SCLK <= 1'b1; end
    24: begin DIN <= r_DAC_DATA[3]; SCLK <= 1'b1; end
    26: begin DIN <= r_DAC_DATA[2]; SCLK <= 1'b1; end
    28: begin DIN <= r_DAC_DATA[1]; SCLK <= 1'b1; end
    30: begin DIN <= r_DAC_DATA[0]; SCLK <= 1'b1; end

    32: SCLK <= 1'b1;
    33: CS_N <= 1'b1;

    default:;

    endcase

end

```

一次转换结束的标志，即 SCLK\_GEN\_CNT[5] && SCLK2X，并产生一个高脉冲的转换完成标志信号 Set\_Done。

```

assign trans_done = (SCLK_GEN_CNT == 33) && SCLK2X;

always@(posedge Clk or negedge Rst_n)
    if(!Rst_n)
        Set_Done <= 1'b0;
    else if(trans_done)
        Set_Done <= 1'b1;
    else
        Set_Done <= 1'b0;

```

## 5.2.6 仿真及板级测试

这里仿真文件需输出几次并行数据，观测串行数据输出 DIN 的状态即可判断是否能驱动正常。这里输出四次数据分别为 C\_AAAh、4\_555h、1\_555h、F\_555h。部分代码如下，只写

出了前两个数据，后面两个可直接复制修改即可。

```
initial begin
    Rst_n = 0; Start = 0; DAC_DATA = 0; #201;
    Rst_n = 1; #200;

    DAC_DATA = 16'hC_AAA; Start = 1; #20;
    Start = 0;
    #200;
    wait(Set_Done);

    #20000;
    DAC_DATA = 16'h4_555; Start = 1; #20;
    Start = 0;
    #200;
    wait(Set_Done);
    $stop;
end
```

开始仿真后，可看出人为控制 DAC\_DATA 数据输入状态正常。

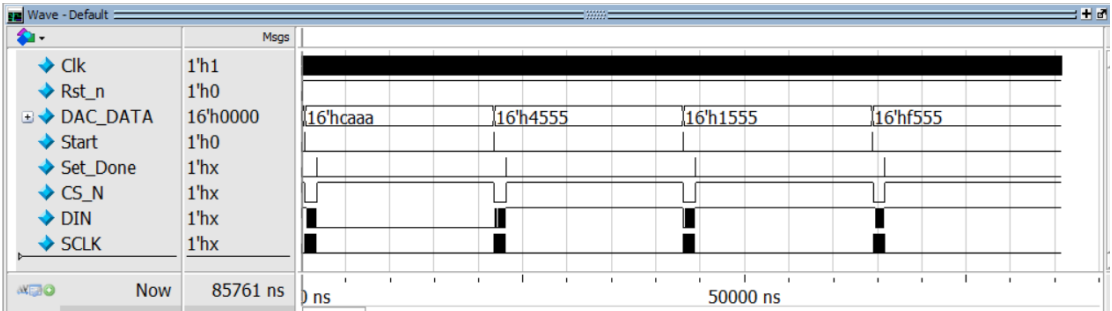


图 24.5 整体功能仿真波形

放大第一个数据传输过程，可以看出正常 1100\_1010\_1010\_1010b，计数器计数到'd32 符合设计要求，且每个传输过程中 CS\_N 为低。传输完成后产生一个时钟周期的 Set\_Done 标志信号。

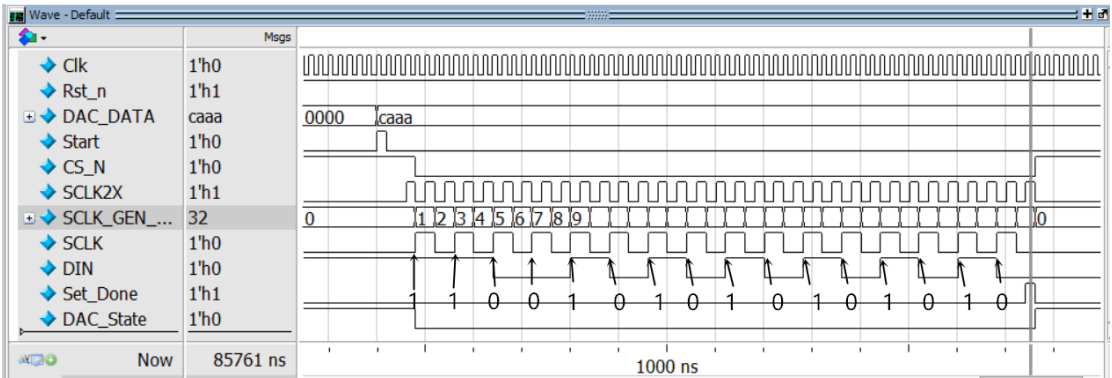


图 24.6 第一次数据传输过程仿真波形

为了再次验证 TLV5618 驱动模块设计的正确性，使用 ISSP 在线调试工具。创建一个 ISSP IP 核，主要配置如图 24.7 所示，详细步骤可参考第 15 章相关内容。

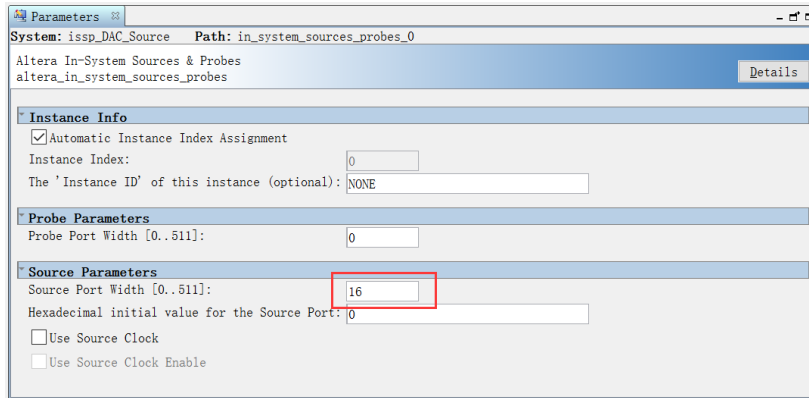


图 24.7 ISSP 主要参数设置

加入工程后新建顶层文件 `DAC_test.v`，并对 ISSP 以及设计好的 TLV5618 进行例化。

在接口时序介绍中指出，TLV5618 有三种更新输出电压方式，下面分别测试这三种电压更新方式。上电复位后两通道输出电压初始值均为 0V。

1. 单独测试 A 通道，依次输入 CFFFh、C7FFh、C1FFh，理论输出电压值应为 4.096、2.048、0.512。可在通道 A 测量输出电压依次为 4.10、2.05、0.51，此时通道 B 电压一直保持 0，电压输出在误差允许范围内。

2. 单独测试 B 通道，依次输入 4FFFh、47FFh、4000h，可在通道 B 测量输出电压依次为 4.10、2.05、0。此时通道 A 电压一直保持 0.51，电压输出在误差允许范围内。

3. 测量 AB 两通道同时更新，首先输入 1FFF 将数据写入通道 B 寄存器，再写入 8FFF 到通道 A 寄存器。这样可以测量出写完后两个通道输出电压会同时变为 4.10。

通过以上三组测试数据的，可以发现 DAC 芯片输出电压数据更新正常。

这样就完成了一个 DAC 模块的设计与仿真验证，基于本节以及 4.3 节即可实现信号发生器，详细内容可以参考第 6 章中的进阶课程 DDS2。