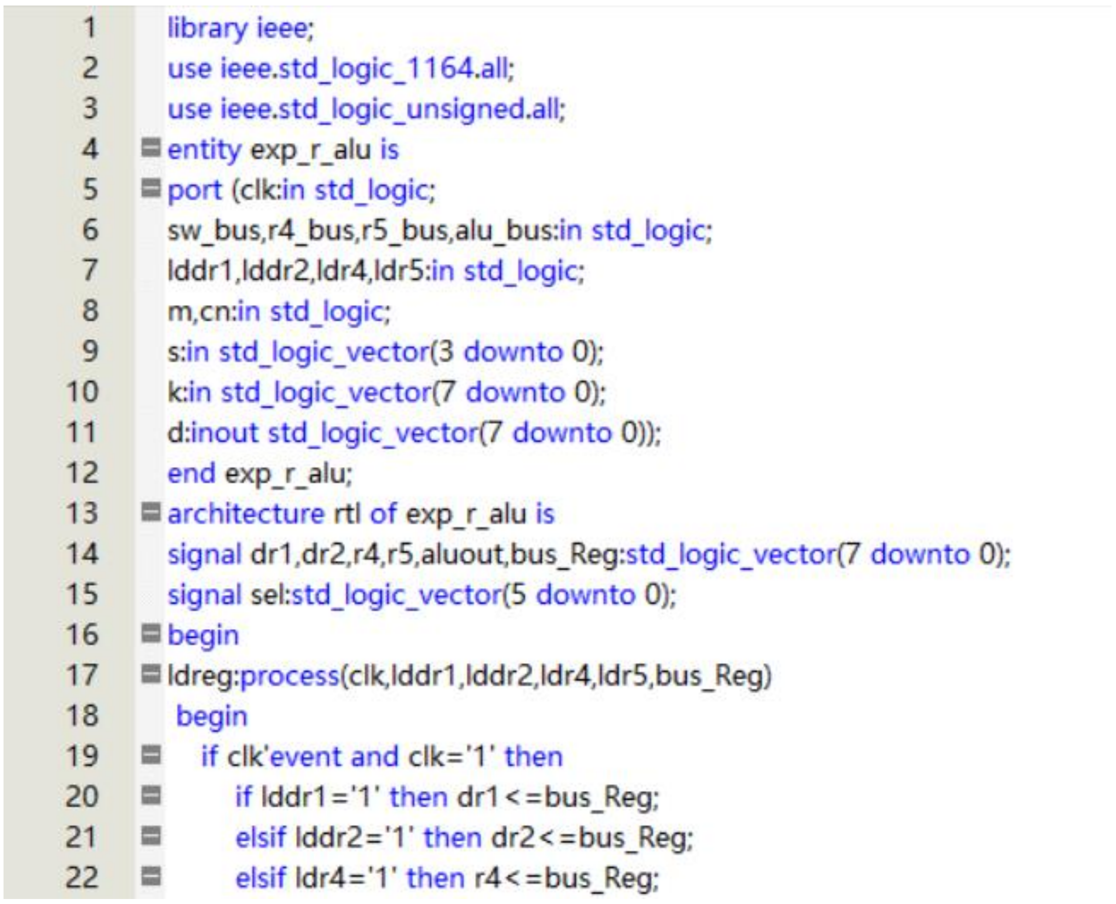
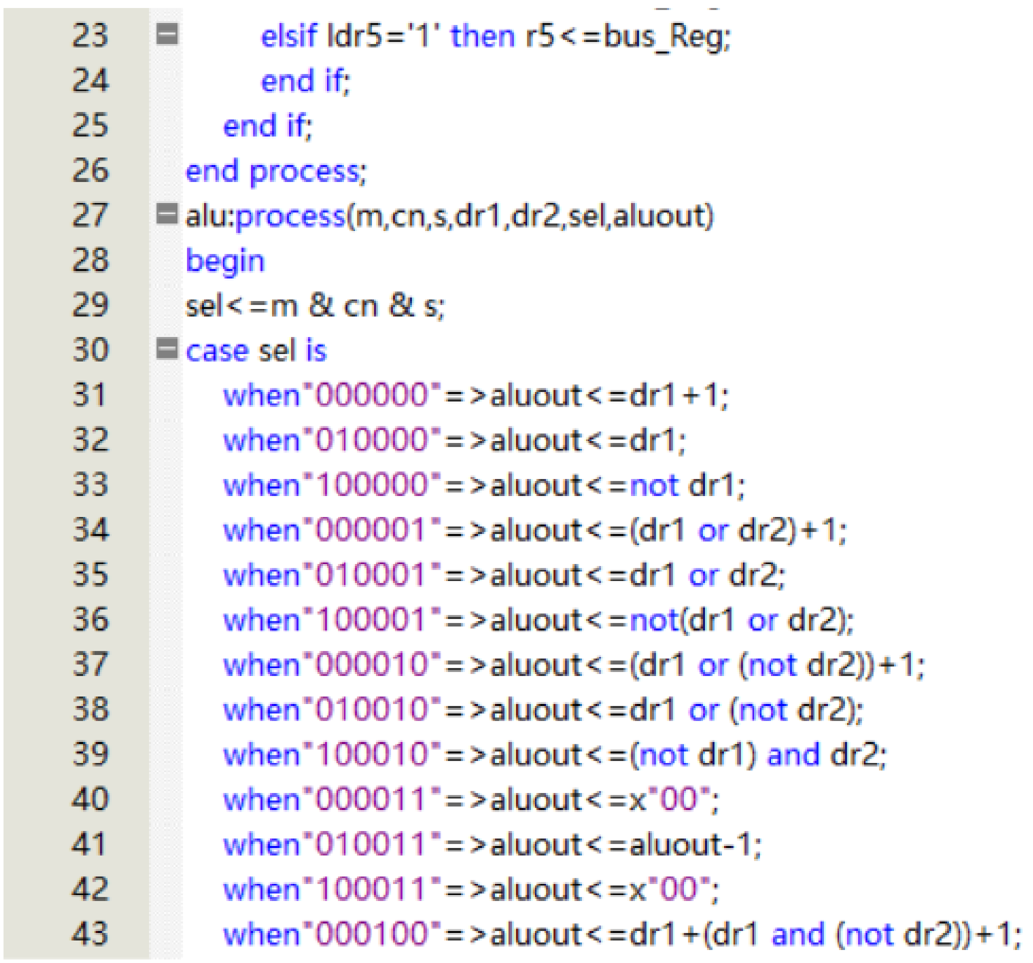
运算器实验日志

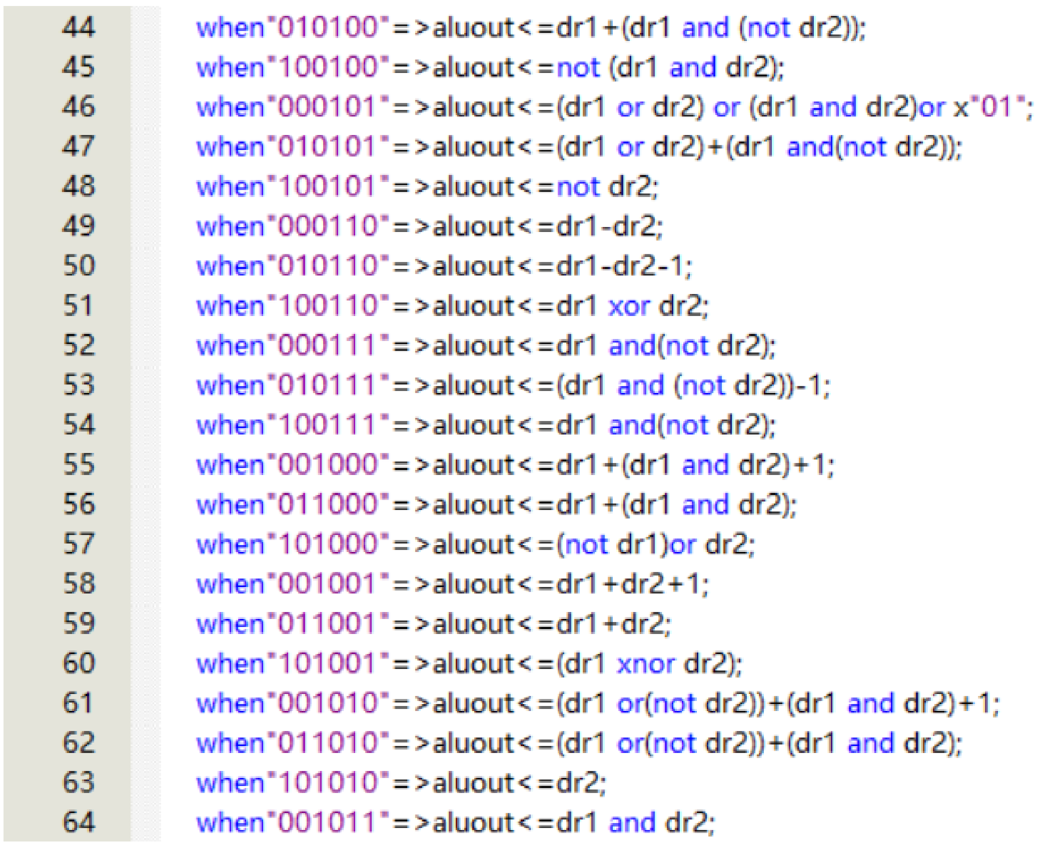
一、实验代码



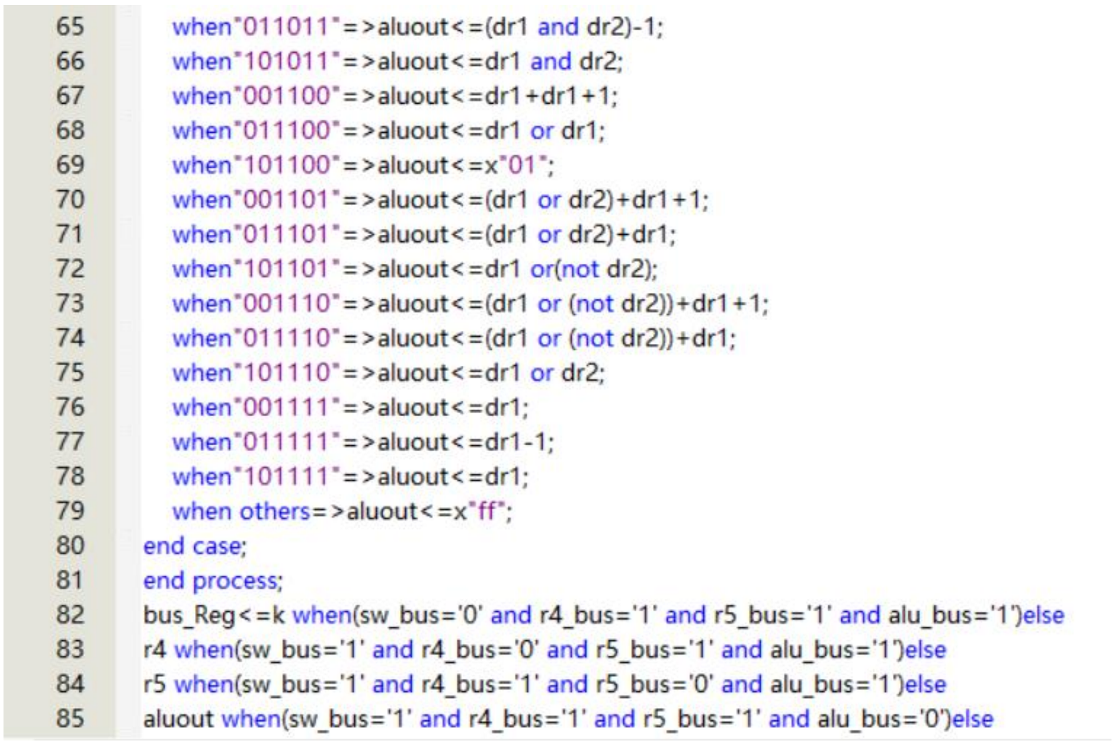
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity exp_r_alu is
port (clk:in std_logic;
sw_bus,r4_bus,r5_bus,alu_bus:in std_logic;
lddr1,lddr2,ldr4,ldr5:in std_logic;
m,cn:in std_logic;
s:in std_logic_vector(3 downto 0);
k:in std_logic_vector(7 downto 0);
d:inout std_logic_vector(7 downto 0));
end exp_r_alu;
architecture rtl of exp_r_alu is
signal dr1,dr2,r4,r5,aluout,bus_Reg:std_logic_vector(7 downto 0);
signal sel:std_logic_vector(5 downto 0);
begin
ldreg:process(clk,lddr1,lddr2,ldr4,ldr5,bus_Reg)
 begin
   if clk'event and clk='1' then
      if lddr1='1' then dr1<=bus_Reg;
      elsif lddr2='1' then dr2<=bus_Reg;
      elsif ldr4='1' then r4<=bus_Reg;
```



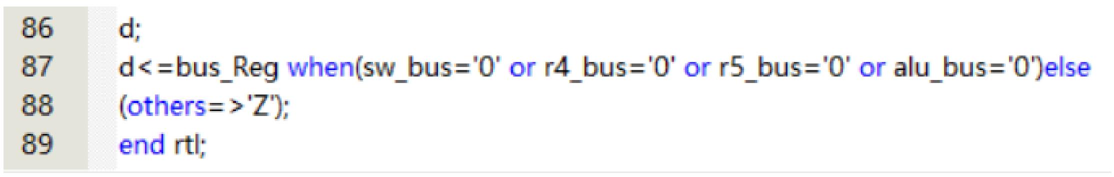
```vhdl
        elsif ldr5='1' then r5<=bus_Reg;
        end if;
    end if;
end process;
alu:process(m,cn,s,dr1,dr2,sel,aluout)
begin
sel<=m & cn & s;
case sel is
    when"000000"=>aluout<=dr1+1;
    when"010000"=>aluout<=dr1;
    when"100000"=>aluout<=not dr1;
    when"000001"=>aluout<=(dr1 or dr2)+1;
    when"010001"=>aluout<=dr1 or dr2;
    when"100001"=>aluout<=not(dr1 or dr2);
    when"000010"=>aluout<=(dr1 or (not dr2))+1;
    when"010010"=>aluout<=dr1 or (not dr2);
    when"100010"=>aluout<=(not dr1) and dr2;
    when"000011"=>aluout<=x"00";
    when"010011"=>aluout<=aluout-1;
    when"100011"=>aluout<=x"00";
    when"000100"=>aluout<=dr1+(dr1 and (not dr2))+1;
```



```vhdl
    when"010100"=>aluout<=dr1+(dr1 and (not dr2));
    when"100100"=>aluout<=not (dr1 and dr2);
    when"000101"=>aluout<=(dr1 or dr2) or (dr1 and dr2)or x"01";
    when"010101"=>aluout<=(dr1 or dr2)+(dr1 and(not dr2));
    when"100101"=>aluout<=not dr2;
    when"000110"=>aluout<=dr1-dr2;
    when"010110"=>aluout<=dr1-dr2-1;
    when"100110"=>aluout<=dr1 xor dr2;
    when"000111"=>aluout<=dr1 and(not dr2);
    when"010111"=>aluout<=(dr1 and (not dr2))-1;
    when"100111"=>aluout<=dr1 and(not dr2);
    when"001000"=>aluout<=dr1+(dr1 and dr2)+1;
    when"011000"=>aluout<=dr1+(dr1 and dr2);
    when"101000"=>aluout<=(not dr1)or dr2;
    when"001001"=>aluout<=dr1+dr2+1;
    when"011001"=>aluout<=dr1+dr2;
    when"101001"=>aluout<=(dr1 xnor dr2);
    when"001010"=>aluout<=(dr1 or(not dr2))+(dr1 and dr2)+1;
    when"011010"=>aluout<=(dr1 or(not dr2))+(dr1 and dr2);
    when"101010"=>aluout<=dr2;
    when"001011"=>aluout<=dr1 and dr2;
```



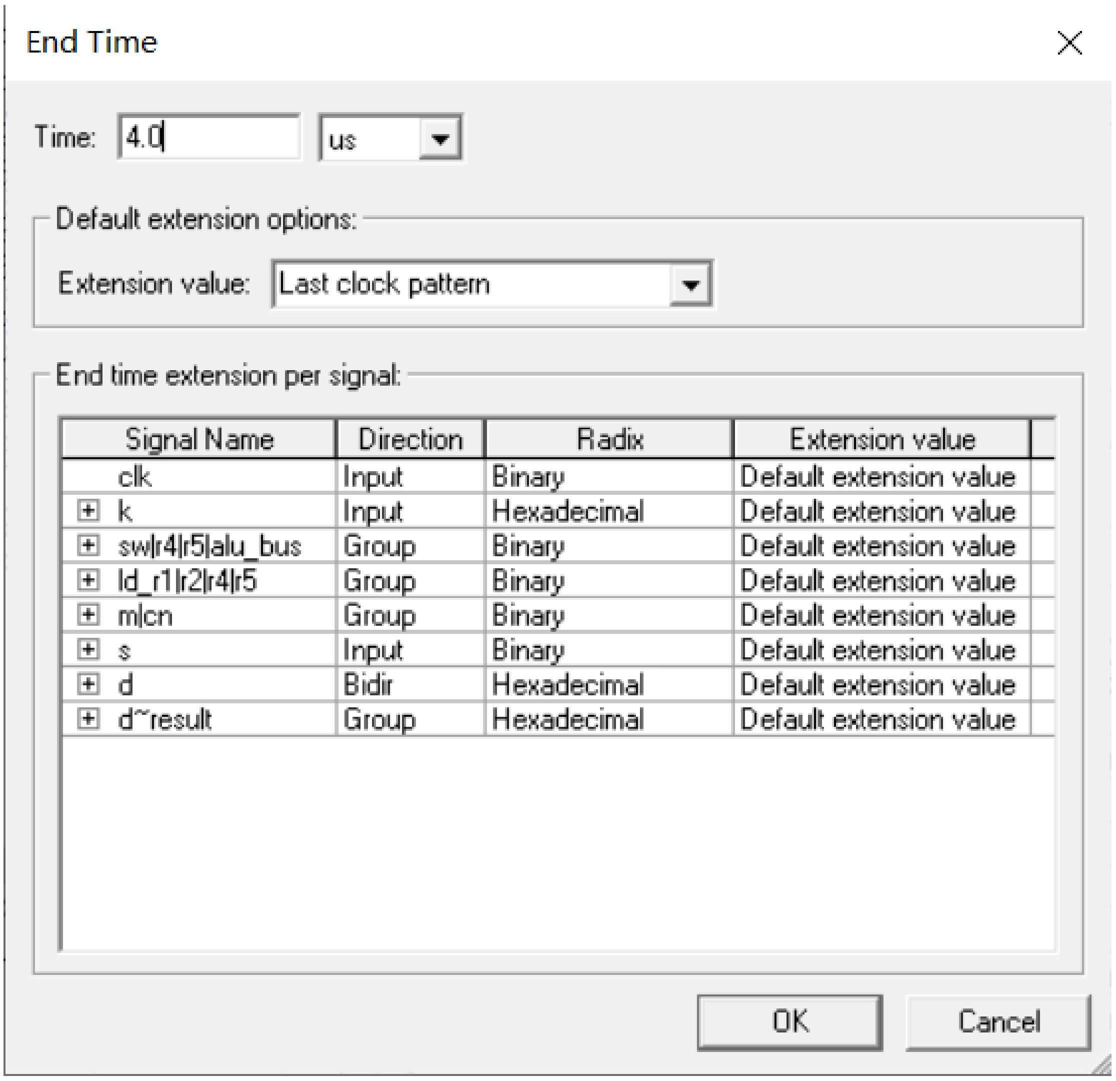
```
    when"011011"=>aluout<=(dr1 and dr2)-1;
    when"101011"=>aluout<=dr1 and dr2;
    when"001100"=>aluout<=dr1+dr1+1;
    when"011100"=>aluout<=dr1 or dr1;
    when"101100"=>aluout<=x"01";
    when"001101"=>aluout<=(dr1 or dr2)+dr1+1;
    when"011101"=>aluout<=(dr1 or dr2)+dr1;
    when"101101"=>aluout<=dr1 or(not dr2);
    when"001110"=>aluout<=(dr1 or (not dr2))+dr1+1;
    when"011110"=>aluout<=(dr1 or (not dr2))+dr1;
    when"101110"=>aluout<=dr1 or dr2;
    when"001111"=>aluout<=dr1;
    when"011111"=>aluout<=dr1-1;
    when"101111"=>aluout<=dr1;
    when others=>aluout<=x"ff";
end case;
end process;
bus_Reg<=k when(sw_bus='0' and r4_bus='1' and r5_bus='1' and alu_bus='1')else
r4 when(sw_bus='1' and r4_bus='0' and r5_bus='1' and alu_bus='1')else
r5 when(sw_bus='1' and r4_bus='1' and r5_bus='0' and alu_bus='1')else
aluout when(sw_bus='1' and r4_bus='1' and r5_bus='1' and alu_bus='0')else
```



```
d;
d<=bus_Reg when(sw_bus='0' or r4_bus='0' or r5_bus='0' or alu_bus='0')else
(others=>'Z');
end rtl;
```

二、实验数据记录

（1）记录波形仿真参数设置（End time 和 Grid size）。


End Time

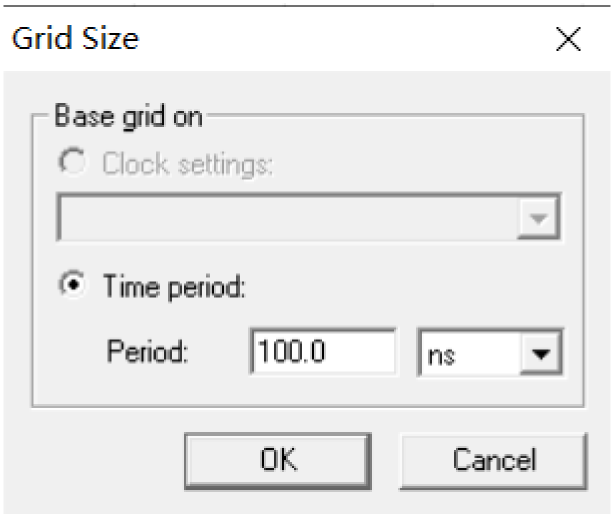
Time: 4.0 us

Default extension options:

Extension value: Last clock pattern

End time extension per signal:

Signal Name	Direction	Radix	Extension value
clk	Input	Binary	Default extension value
⊞ k	Input	Hexadecimal	Default extension value
⊞ sw|r4|r5|alu_bus	Group	Binary	Default extension value
⊞ ld_r1|r2|r4|r5	Group	Binary	Default extension value
⊞ m|cn	Group	Binary	Default extension value
⊞ s	Input	Binary	Default extension value
⊞ d	Bidir	Hexadecimal	Default extension value
⊞ d~result	Group	Hexadecimal	Default extension value


Grid Size

Base grid on

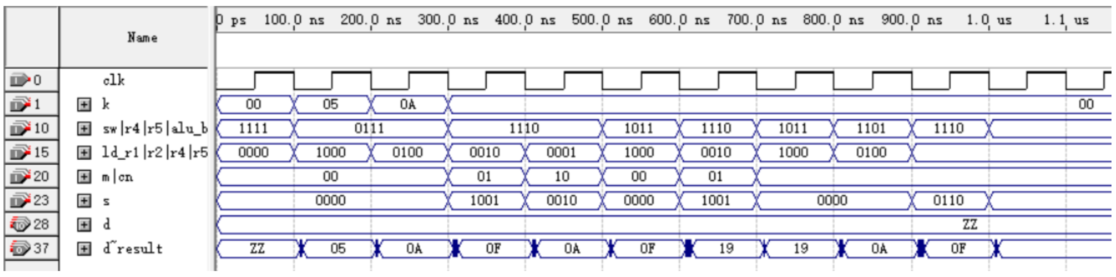
Clock settings:

[✓ Time period:]

Period: 100.0 ns

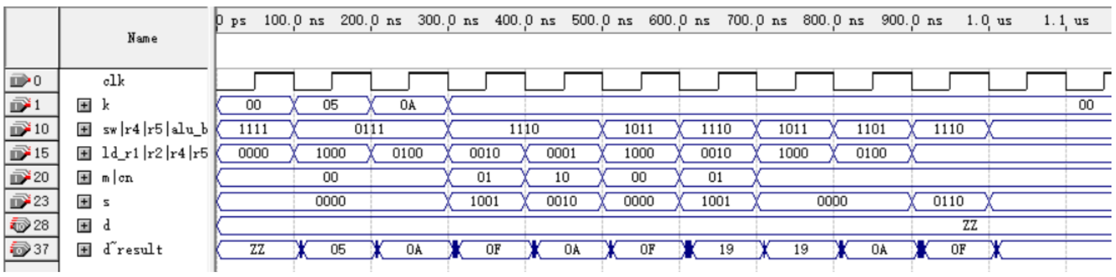
（2）记录芯片设置及引脚设置。

（3）记录电路初始状态时 input 输入信号设置。


Name
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us 1.1 us
clk
k 00 05 0A 00
sw|r4|r5|alu_b 1111 0111 1110 1011 1110 1011 1101 1110
ld_r1|r2|r4|r5 0000 1000 0100 0010 0001 1000 0010 1000 0100
m|cn 00 01 10 00 01
s 0000 1001 0010 0000 1001 0000 0110
d ZZ
d~result ZZ 05 0A 0F 0A 0F 19 19 0A 0F

k 置为 00，sw_bus,r4_bus,r5_bus,alu_bus 低电平有效，置为 1111，ldr1、ldr2、ldr4、ldr5 高电平有效，置为 0000，m 置为 0, cn 置为 0,s 置为 0000。

（4）记录全部实验内容中出现的数据。


Name
0 ps 100.0 ns 200.0 ns 300.0 ns 400.0 ns 500.0 ns 600.0 ns 700.0 ns 800.0 ns 900.0 ns 1.0 us 1.1 us
clk
k 00 05 0A 00
sw|r4|r5|alu_b 1111 0111 1110 1011 1110 1011 1101 1110
ld_r1|r2|r4|r5 0000 1000 0100 0010 0001 1000 0010 1000 0100
m|cn 00 01 10 00 01
s 0000 1001 0010 0000 1001 0000 0110
d ZZ
d~result ZZ 05 0A 0F 0A 0F 19 19 0A 0F

0~100ns,设置初始状态；

100~200ns,将外部输入 05 传入总线，再写入 r1；

200~300ns,将外部输入 0A 传入总线，再写入 r2；

300~400ns,将 r1 和 r2 中的数据相加，将结果 0F 传入总线再写入 r4;

400~500ns, 将 r1 和 r2 中的数据作((not A)and B)运算，将结果 0A 传入总线再写入 r5;

500~600ns,将 r4 数据写入 r1;

600~700ns, 将 r1 和 r2 中的数据相加，将结果 19H 传入总线再写入 r4；

700~800ns, 将 r4 数据再次写入 r1；

800~900ns, 将 r5 数据写入 r2；

900~1000ns, 将 r1 和 r2 中的数据相减，将结果 0F 传入总线；

1us~1.1us,停机。

三、思考题

1）存入 DR1 或 DR2 的数据如何在总线上显示？

通过 m，cn，s 控制信号控制来输出到总线，如想要展示 DR1 的值，需要将 m,cn,s 设置为 101111 即可以在总线上输出；如想要展示 DR2 的值，需要将 m,cn,s 设置为 101010 即可以在总线上输出。

2）复合运算时，ALU 运算出的中间结果为什么不能直接存入 DR1 或 DR2?

DR1 和 DR2 存储的是最初的原始数据，如果将中间结果继续存到 DR1 或 DR2 中，那么在后续想要调用最初的原始数据将会无法调用。

3）计算机中的负数如何表示？

通过符号位的 0 和 1 来表示正数和负数。

4）74181 的功能表中运算+与“加”的区别是什么？

+表示逻辑加法运算，“加”表示算数加法运算。

5）exp_r_alu.vhd 中并置运算符&主要作用是什么？

用于连接数据，如 a=“01”，b=“10”，那么 c=a&b 的结果就是“0110”。

6）exp_s_alu.vhd 代码中为什么要调用 ieee.std_logic_unsigned 库？

为了使用+、-等运算符来简化代码。

7）VHDL 语言中如何表示十六进制格式数据？

通过 x“0”来表示，如 x“0”即表示十六进制里面的 0。

8）试述使用 8 位运算器如何实现复合运算？

通过 DR1 和 DR2 暂存器来存储中间结果，使用多个时钟周期来实现复杂的复合运算。