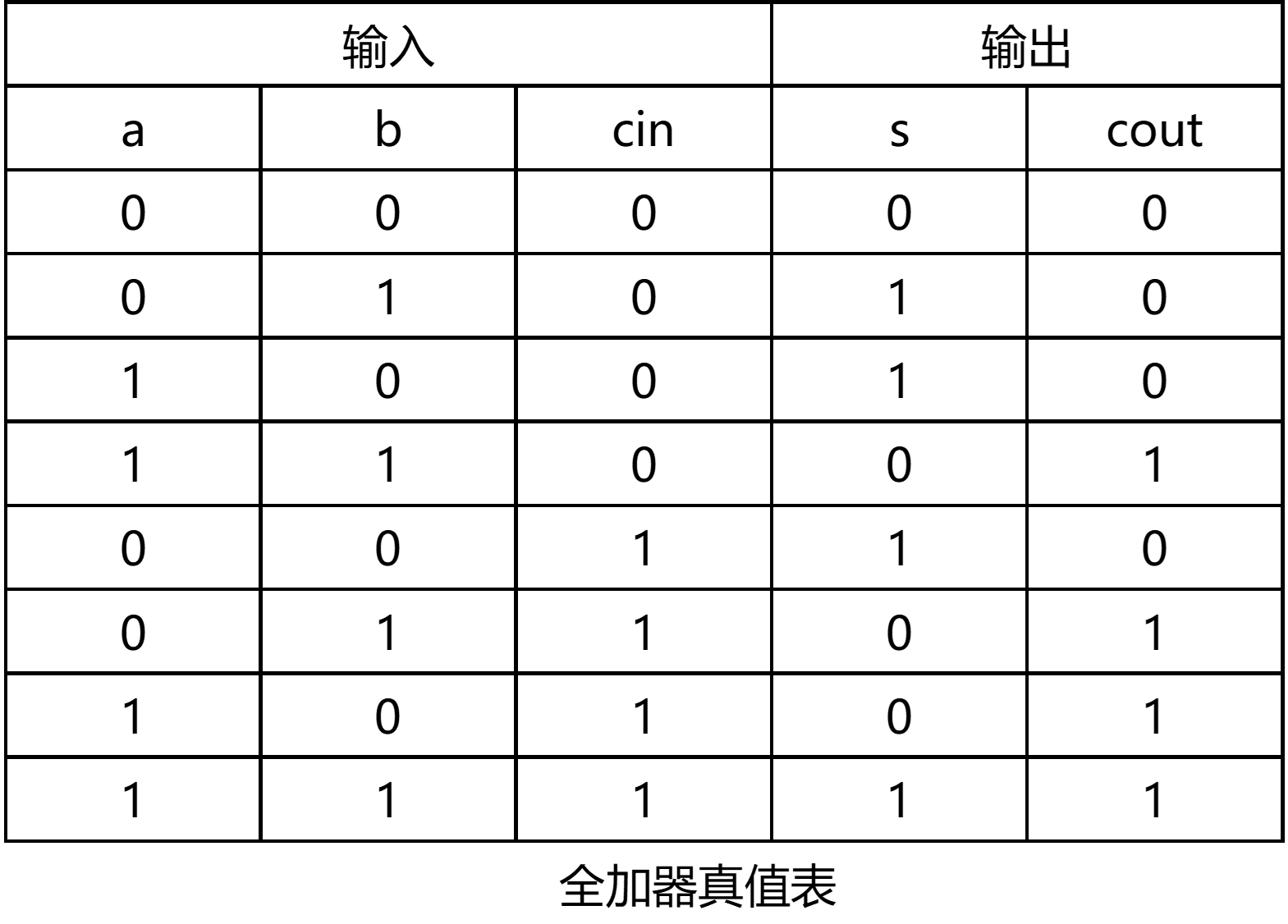
**四位全加器**

**一、设计分析：**

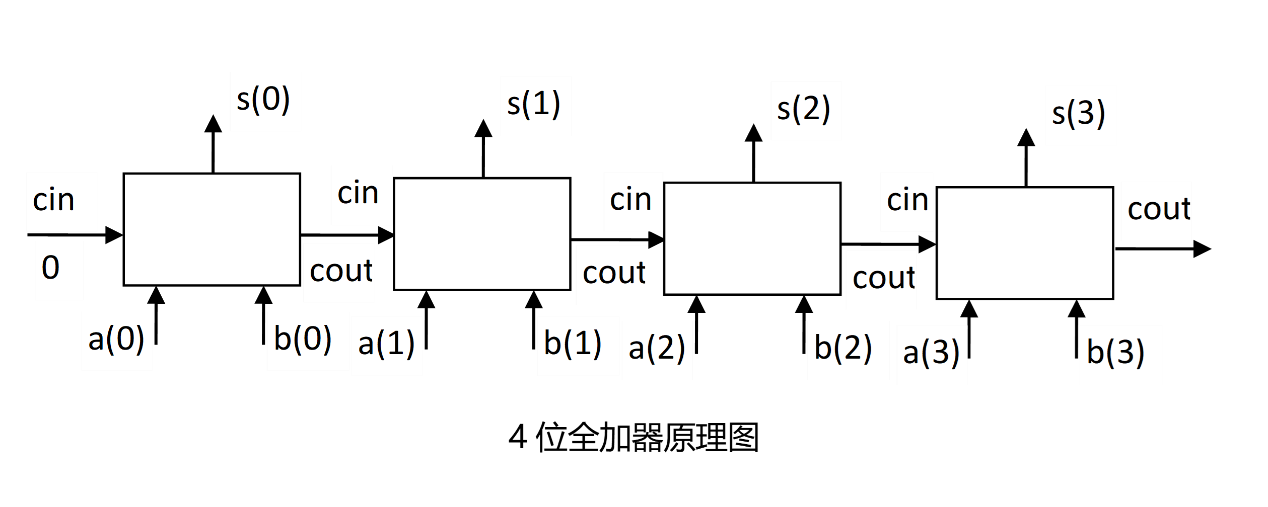
全加器是能进行加数、被加数和低位来的进位信号相加，并根据求和结果给出该位的进位信号的加法电路。

真值表如图所示：



根据真值表可得出表达式：

根据以上表达式，可以用数据流方式设计出1位全加器。用串行进位来设计4位全加器。先设计4个1位的全加器，然后将低位的进位输出与高位的进位输入相连，将要进行加法运算的两个4位数的每一位分别作为每一个1位全加器的输入，进行加法运算，所有的1位全加器的输出组成一个4位数，即输入的两个4位数之和，最高位的全加器产生的进位输出即两个4位数求和的进位输出。



**二、代码设计：**

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY adder1 IS --1位全加器设计

PORT(a,b,cin:IN STD\_LOGIC;

s,cout:OUT STD\_LOGIC);

END adder1;

ARCHITECTURE dataflow OF adder1 IS --用数据流方式设计1位全加器

SIGNAL tmp:STD\_LOGIC; --用tmp表示a⊕b

BEGIN

tmp<=a XOR b AFTER 10 ns;

s<=tmp XOR cin AFTER 10 ns;

cout<=(a AND b)OR(tmp AND cin) AFTER 20 ns;

END dataflow;

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY adder4 IS --4位全加器设计

PORT(a,b:IN STD\_LOGIC\_VECTOR(3 DOWNTO 0);

cin:IN STD\_LOGIC;

s:OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0);

cout:OUT STD\_LOGIC);

END adder4;

ARCHITECTURE structural OF adder4 IS --用结构化描述风格设计4位全加器

COMPONENT adder1

PORT(a,b,cin:IN STD\_LOGIC;

s,cout:OUT STD\_LOGIC);

END COMPONENT;

SIGNAL x,y,z:STD\_LOGIC;

FOR u1,u2,u3,u4:adder1 USE ENTITY WORK.adder1(dataflow);

BEGIN

u1:adder1 PORT MAP(a(0),b(0),cin,s(0),x);

u2:adder1 PORT MAP(a(1),b(1),x,s(1),y);

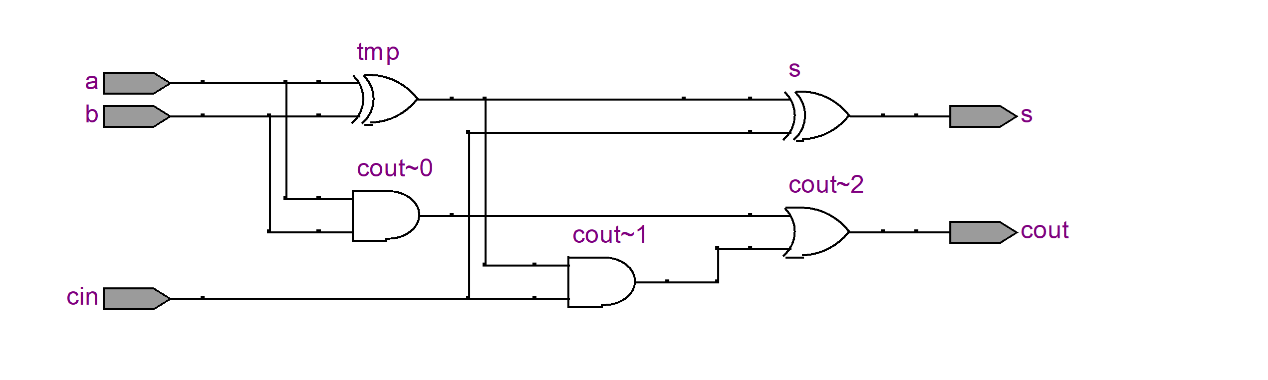
u3:adder1 PORT MAP(a(2),b(2),y,s(2),z);

u4:adder1 PORT MAP(a(3),b(3),z,s(3),cout);

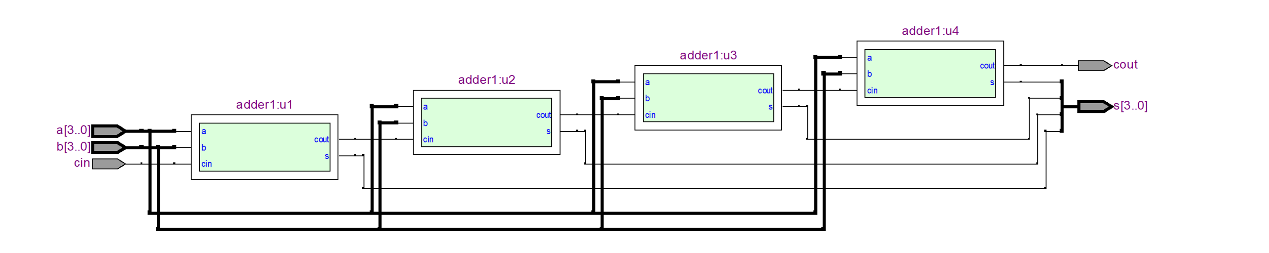
END structural;

**三、逻辑图**

1位全加器仿真逻辑图：



4位全加器逻辑图：



**四、波形仿真**

测试数据：

|  |  |  |
| --- | --- | --- |
| Cin\_a | Cin\_b | Cout |
| 1001 | 0101 | 0000 1110 |
| 0111 | 1010 | 0001 0001 |

4位全加器仿真：

