**信息科学与工程学院**

**2019－2020学年第二学期**

实 验 报 告

课程名称： 电子设计自动化

实验名称： 实验一 加法器设计

专 业 班 级 通信工程 二班

学 生 学 号 201800121050

学 生 姓 名 孟麟芝

实 验 时 间 2020年4月

实验报告

【实验目的】

1.掌握使用VHDL语言进行硬件设计的基本方法

2.掌握基本的Quartus Ⅱ使用方法

3.学会testbench的编写，掌握波形仿真的基本方法

【实验要求】

1.设计一个16位二进制全加器模块。

2.用层次化设计方法，设计一个16位二进制全加器模块。

3.设计一个16位超前进位二进制全加器模块。

4.设计一个8位8421-BCD码全加器模块。（\*）

【实验具体内容】

### 【第一个实验】

1.实验原理

这一设计使用非层次化的设计，较为简便，直接使用quartus内置的加法运算功能即可完成输出。即定义一个17维向量，将A+B+CIN得结果存于其中，低16位赋给输出，最高位赋给进位信号即可。

2.设计模块代码（Design Block）

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER16\_A IS

PORT (A : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

B : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

CIN : IN STD\_LOGIC;

SUM : OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0);

COUT : OUT STD\_LOGIC);

END ENTITY ADDER16\_A;

ARCHITECTURE BHV OF ADDER16\_A IS

SIGNAL DATA :STD\_LOGIC\_VECTOR(16 DOWNTO 0);

BEGIN

DATA <= ('0'&A) + ('0'&B) + ("0000000000000000"&CIN);

COUT <= DATA(16);

SUM <= DATA(15 DOWNTO 0);

END ARCHITECTURE BHV;

3.激励模块代码（Test Bench）

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER16\_A\_vhd\_tst IS

END ADDER16\_A\_vhd\_tst;

ARCHITECTURE ADDER16\_A\_arch OF ADDER16\_A\_vhd\_tst IS

SIGNAL A1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

SIGNAL B1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

SIGNAL CIN1 : STD\_LOGIC;

SIGNAL COUT1 : STD\_LOGIC;

SIGNAL SUM1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

COMPONENT ADDER16\_A

PORT (

A : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

B : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

CIN : IN STD\_LOGIC;

COUT : OUT STD\_LOGIC;

SUM : OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0)

);

END COMPONENT;

BEGIN

i1 : ADDER16\_A

PORT MAP (

A => A1,

B => B1,

CIN => CIN1,

COUT => COUT1,

SUM => SUM1 );

A1 <= "1100111100001111","1000000000000000" AFTER 100 NS, "0111111111111111" AFTER 800 NS;

B1 <= "1100111100001100","1000000000001100" AFTER 100 NS, "0111111111111111" AFTER 800 NS;

CIN1 <= '1','0' AFTER 500 NS, '1' AFTER 800NS;

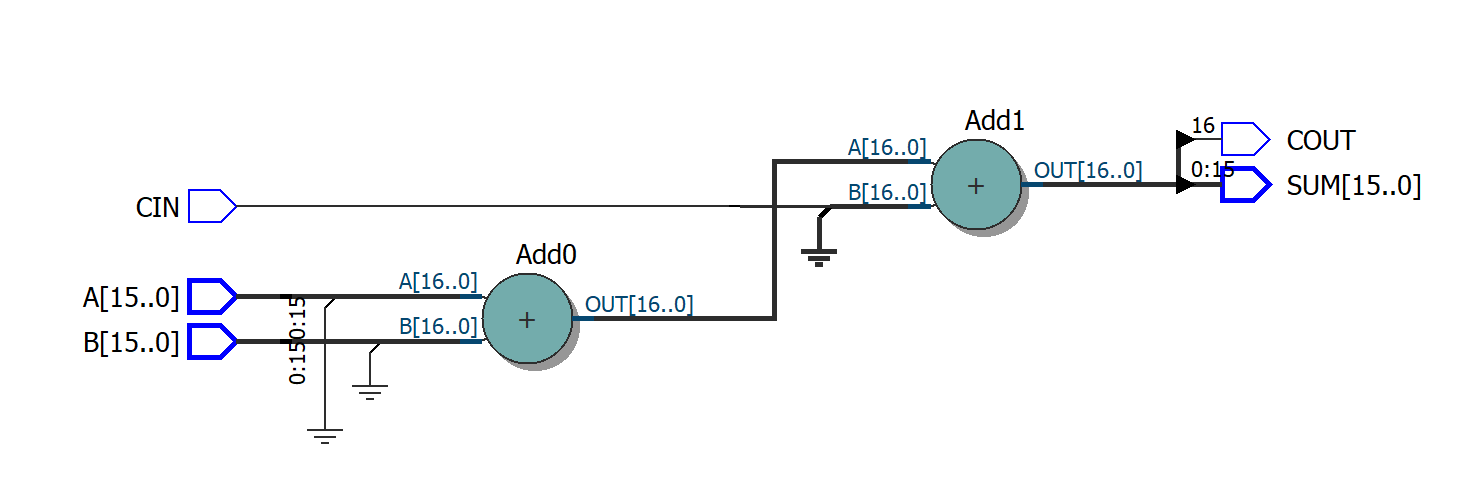
END ADDER16\_A\_arch;

4.仿真波形图



（图中的数据使用的是四位十六进制显示）

5.门级电路图



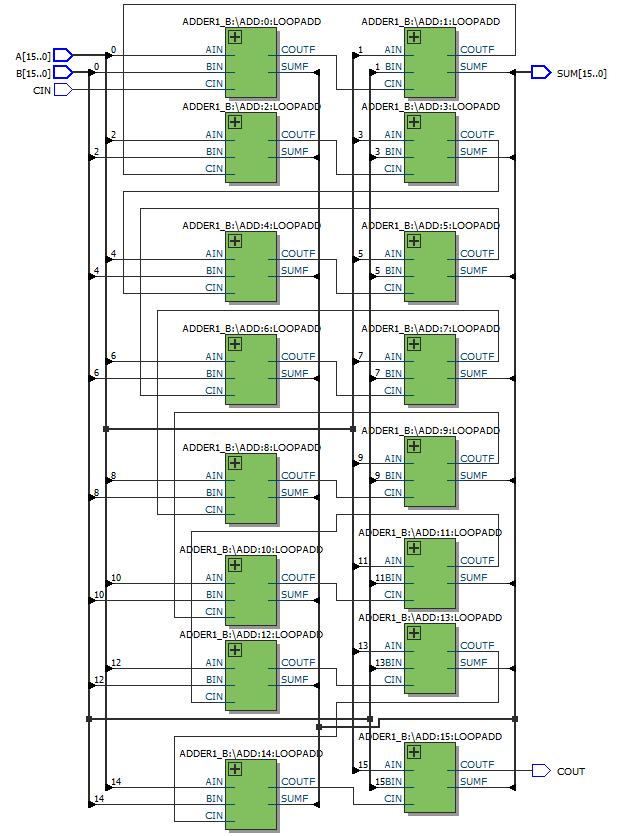
6.结果分析与思考

这种非层次化的实现方法较为简便，而对硬件逻辑资源的占用较多，但由于此电路较为简单，并无大碍。

### 【第二个实验】

1.实验原理

该设计通过层次化设计的方法，将两个半加器与一个或门连接成一个一位全加器，将16个一位全加器级联即可构成16位全加器，具体的电路图如下



2.设计模块代码（Design Block）

（1）半加器代码：

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY HADDER1 IS

PORT(A,B:IN STD\_LOGIC;

CO:OUT STD\_LOGIC;

SO:OUT STD\_LOGIC

);

END HADDER1;

ARCHITECTURE ONE OF HADDER1 IS

BEGIN

SO<=A XOR B;

CO<=A AND B;

END ONE;

（2）或门代码

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY GATE\_OR IS

PORT(A,B:IN STD\_LOGIC;

C:OUT STD\_LOGIC

);

END GATE\_OR;

ARCHITECTURE ONE OF GATE\_OR IS

BEGIN

C<=A OR B;

END ONE;

（5）一位全加器代码

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER1\_B IS

PORT(AIN:IN STD\_LOGIC;

BIN:IN STD\_LOGIC;

CIN:IN STD\_LOGIC;

COUTF:OUT STD\_LOGIC;

SUMF:OUT STD\_LOGIC

);

END ADDER1\_B;

ARCHITECTURE ONE OF ADDER1\_B IS

COMPONENT HADDER1

PORT(A:IN STD\_LOGIC;

B:IN STD\_LOGIC;

CO:OUT STD\_LOGIC;

SO:OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT GATE\_OR

PORT(A:IN STD\_LOGIC;

B:IN STD\_LOGIC;

C:OUT STD\_LOGIC

);

END COMPONENT;

SIGNAL D,E,F:STD\_LOGIC;

BEGIN

U1:HADDER1 PORT MAP(A=>AIN,B=>BIN,CO=>D,SO=>E);

U2:HADDER1 PORT MAP(A=>E,B=>CIN,CO=>F,SO=>SUMF);

U3:GATE\_OR PORT MAP(A=>D,B=>F,C=>COUTF);

END ONE;

（4）十六位全加器代码

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER16\_B IS

PORT(A:IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

B:IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

CIN:IN STD\_LOGIC;

SUM:OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0);

COUT:OUT STD\_LOGIC

);

END;

ARCHITECTURE ONE OF ADDER16\_B IS

COMPONENT ADDER1\_B

PORT(AIN:IN STD\_LOGIC;

BIN:IN STD\_LOGIC;

CIN:IN STD\_LOGIC;

COUTF:OUT STD\_LOGIC;

SUMF:OUT STD\_LOGIC

);

END COMPONENT;

SIGNAL DATA:STD\_LOGIC\_VECTOR(16 DOWNTO 0);

BEGIN

DATA(0)<=CIN;

ADD:FOR I IN 0 TO 15 GENERATE

LOOPADD:ADDER1\_B PORT MAP(AIN=>A(I),

BIN=>B(I),

CIN=>DATA(I),

COUTF=>DATA(I+1),

SUMF=>SUM(I)

);

END GENERATE ADD;

COUT<=DATA(16);

END ONE;

3.激励模块代码（Test Bench）

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER16\_B\_vhd\_tst IS

END ADDER16\_B\_vhd\_tst;

ARCHITECTURE ADDER16\_B\_arch OF ADDER16\_B\_vhd\_tst IS

SIGNAL A1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

SIGNAL B1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

SIGNAL CIN1 : STD\_LOGIC;

SIGNAL COUT1 : STD\_LOGIC;

SIGNAL SUM1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

COMPONENT ADDER16\_B

PORT (

A : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

B : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

CIN : IN STD\_LOGIC;

COUT : OUT STD\_LOGIC;

SUM : OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0)

);

END COMPONENT;

BEGIN

i1 : ADDER16\_B

PORT MAP (

A => A1,

B => B1,

CIN => CIN1,

COUT => COUT1,

SUM => SUM1

);

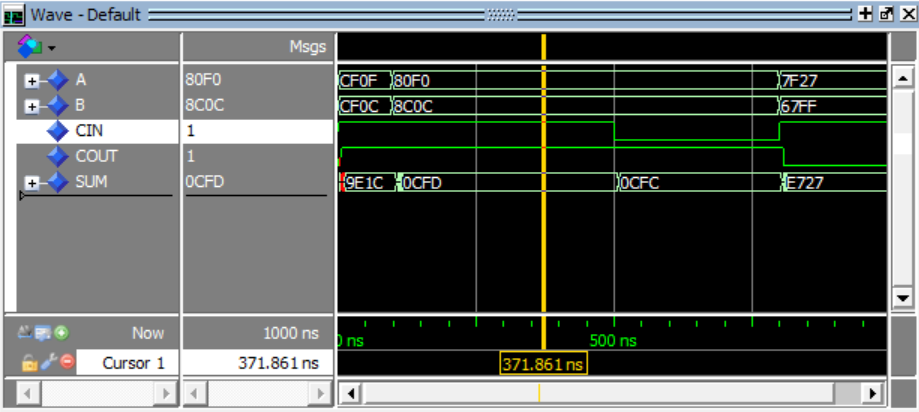
A1 <= "1100111100001111","1000000011110000" AFTER 100 NS, "0111111100100111" AFTER 800 NS;

B1 <= "1100111100001100","1000110000001100" AFTER 100 NS, "0110011111111111" AFTER 800 NS;

CIN1 <= '1','0' AFTER 500 NS, '1' AFTER 800NS;

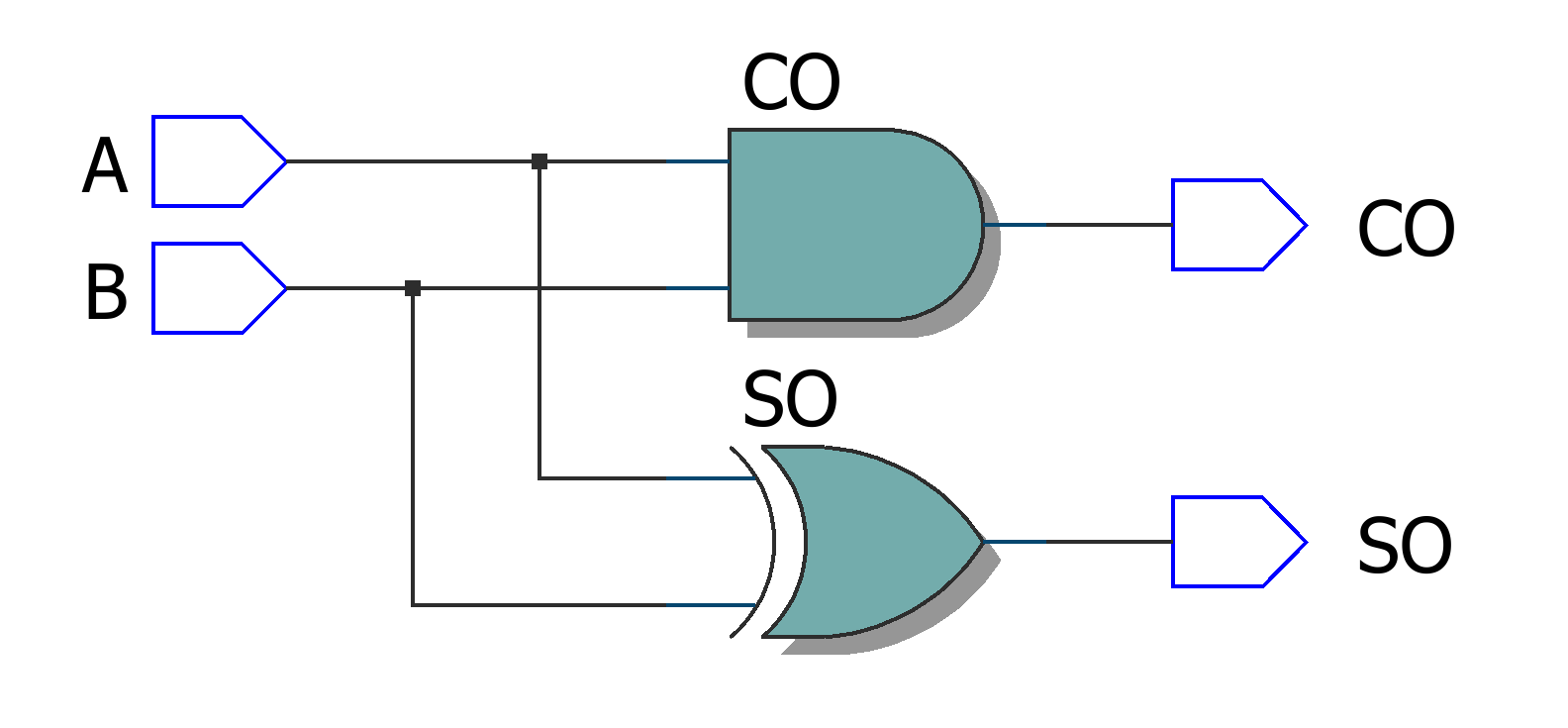
END ADDER16\_B\_arch;

4.仿真波形图

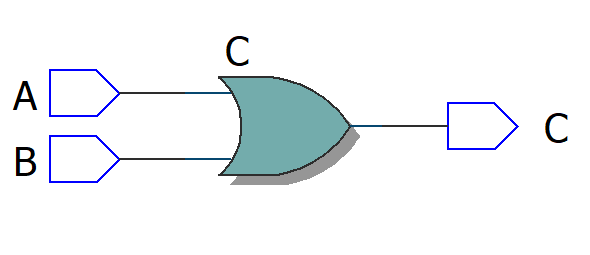


5.门级电路图

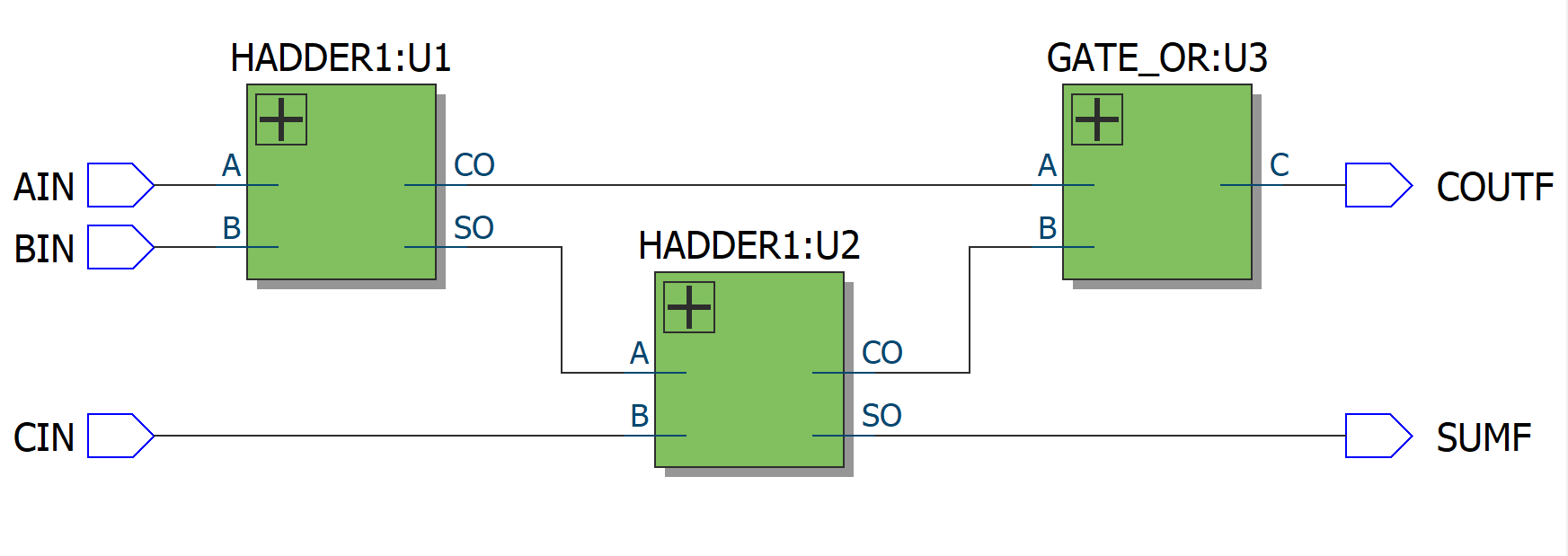
（1）一位半加器



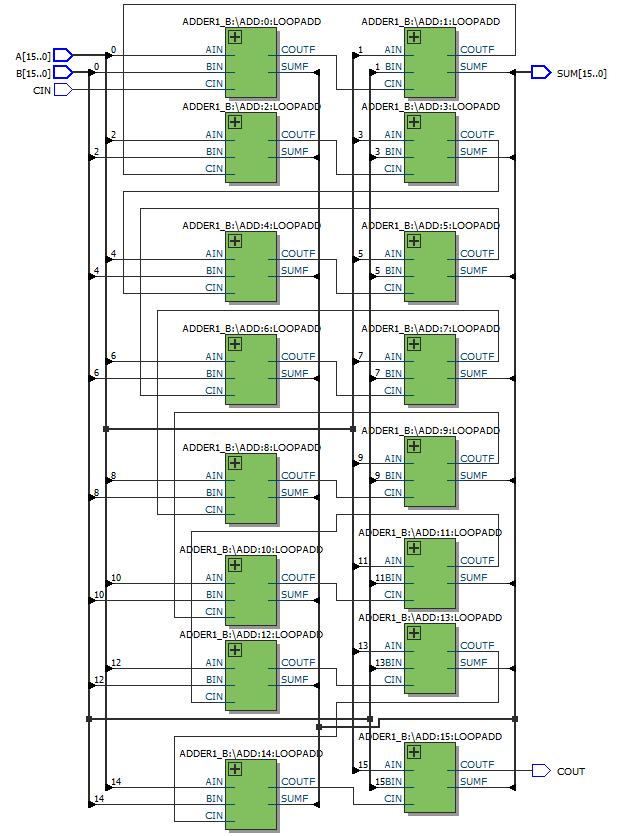
（2）或门



（3）一位全加器



（4）十六位全加器



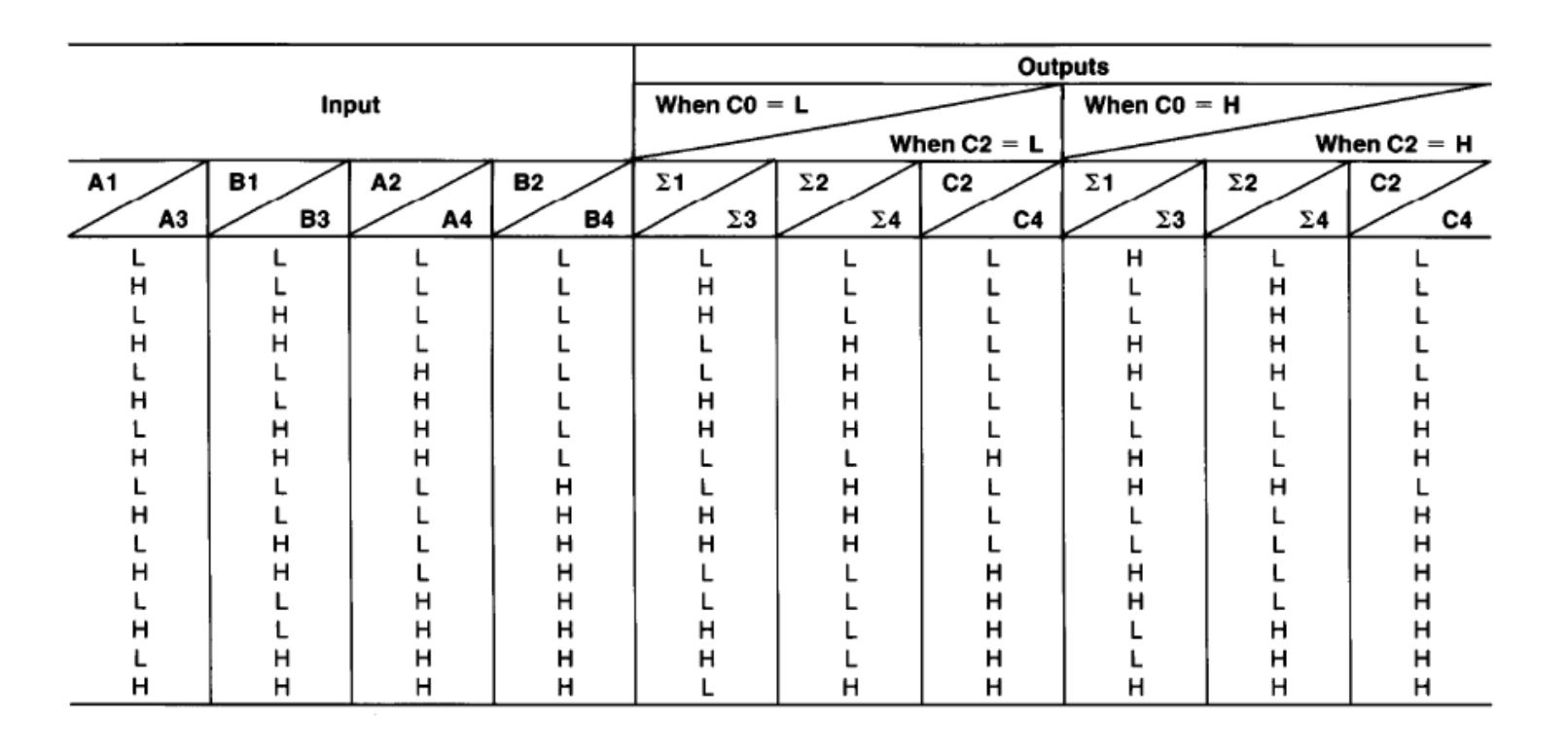
6.结果分析与思考

如图所见，该电路能实现相应的功能，而由于级联较多，且没有进行超前进位，其速度有一定的限制，出现输出时有抖动

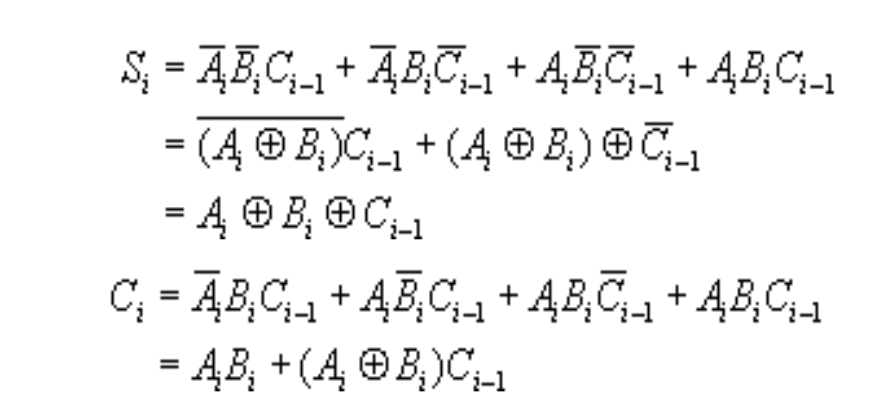
### 【第三个实验】

1.实验原理

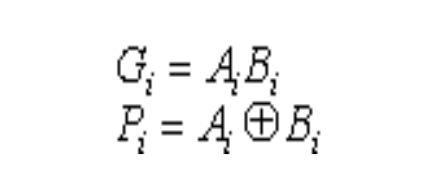
如图为四位超前全加器的真值表

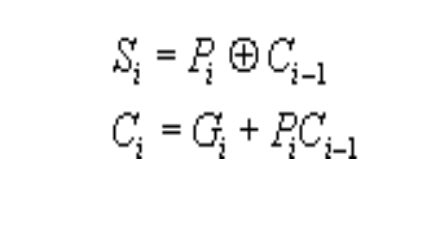


逻辑原理：

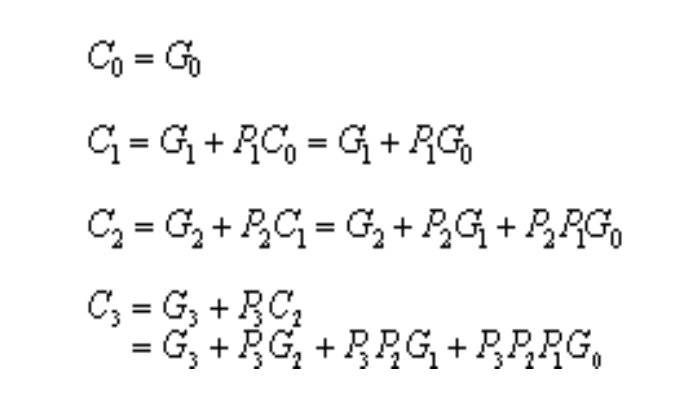


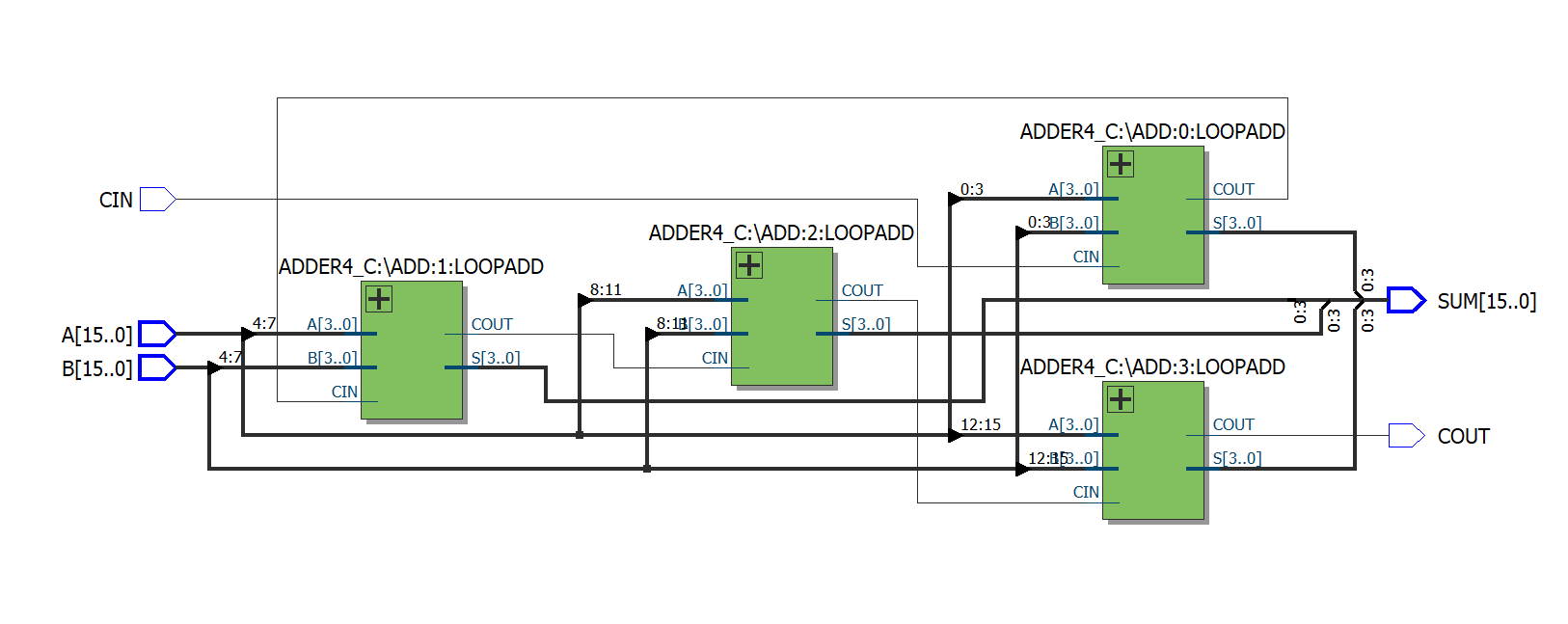
定义两个中间变量方便运算,则有：





则可得到各位的超前进位信号为：





得到一个四位超前二进制全加器后，需要将四个全加器级联，要注意两个四位全加器之间的进位信号仍然是同步的，并没有超前的关系

2.设计模块代码（Design Block）

（1）四位超前二进制全加器代码

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

ENTITY ADDER4\_C IS

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 ENTITY ADDER4\_C;

ARCHITECTURE BHV OF ADDER4\_C IS

SIGNAL SA,SB,SC,SS,ST,SG : STD\_LOGIC\_VECTOR(3 DOWNTO 0);

SIGNAL SCIN,SCOUT : STD\_LOGIC;

BEGIN

SA<=A;

SB<=B;

SCIN<=CIN;

ST(0)<= SA(0) XOR SB(0);--P0

SG(0)<= SA(0) AND SB(0);--G0

ST(1)<= SA(1) XOR SB(1);--P1

SG(1)<= SA(1) AND SB(1);--G1

ST(2)<= SA(2) XOR SB(2);--P2

SG(2)<= SA(2) AND SB(2);--G2

ST(3)<= SA(3) XOR SB(3);--P3

SG(3)<= SA(3) AND SB(3);--G3

SC(0)<= SG(0) OR (ST(0) AND SCIN);--C0

SC(1)<= SG(1) OR (ST(1) AND (SG(0) OR (ST(0) AND SCIN)));--C1

SC(2)<= SG(2) OR (ST(2) AND (SG(1) OR (ST(1) AND (SG(0) OR(ST(0) AND SCIN)))));--C2

SC(3)<= SG(3) OR (ST(3) AND (SG(2) OR (ST(2) AND (SG(1) OR (ST(1) AND (SG(0) OR (ST(0) AND SCIN)))))));--C3

SS(0)<= ST(0) XOR SCIN ;--S0

SS(1)<= ST(1) XOR SC(0);--S1

SS(2)<= ST(2) XOR SC(1);--S2

SS(3)<= ST(3) XOR SC(2);--S3

S <= SS;

COUT <= SC(3);

END ARCHITECTURE BHV;

（2）十六位超前二进制全加器代码

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER16\_C IS

PORT(A:IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

B:IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

CIN:IN STD\_LOGIC;

SUM:OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0);

COUT:OUT STD\_LOGIC

);

END;

ARCHITECTURE ONE OF ADDER16\_C IS

COMPONENT ADDER4\_C

PORT(A:IN STD\_LOGIC\_VECTOR(3 DOWNTO 0);

B:IN STD\_LOGIC\_VECTOR(3 DOWNTO 0);

CIN:IN STD\_LOGIC;

COUT:OUT STD\_LOGIC;

S:OUT STD\_LOGIC\_VECTOR(3 DOWNTO 0)

);

END COMPONENT;

SIGNAL DATA:STD\_LOGIC\_VECTOR(4 DOWNTO 0);

BEGIN

DATA(0)<=CIN;

ADD:FOR I IN 0 TO 3 GENERATE

LOOPADD:ADDER4\_C PORT MAP(A=>A(4\*I+3 DOWNTO 4\*I),

B=>B(4\*I+3 DOWNTO 4\*I),

CIN=>DATA(I),

COUT=>DATA(I+1),

S=>SUM(4\*I+3 DOWNTO 4\*I)

);

END GENERATE ADD;

COUT<=DATA(4);

END ONE;

3.激励模块代码（Test Bench）

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY ADDER16\_C\_vhd\_tst IS

END ADDER16\_C\_vhd\_tst;

ARCHITECTURE ADDER16\_C\_arch OF ADDER16\_C\_vhd\_tst IS

SIGNAL A1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

SIGNAL B1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

SIGNAL CIN1 : STD\_LOGIC;

SIGNAL COUT1 : STD\_LOGIC;

SIGNAL SUM1 : STD\_LOGIC\_VECTOR(15 DOWNTO 0);

COMPONENT ADDER16\_C

PORT (

A : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

B : IN STD\_LOGIC\_VECTOR(15 DOWNTO 0);

CIN : IN STD\_LOGIC;

COUT : OUT STD\_LOGIC;

SUM : OUT STD\_LOGIC\_VECTOR(15 DOWNTO 0)

);

END COMPONENT;

BEGIN

i1 : ADDER16\_C

PORT MAP (

A => A1,

B => B1,

CIN => CIN1,

COUT => COUT1,

SUM => SUM1

);

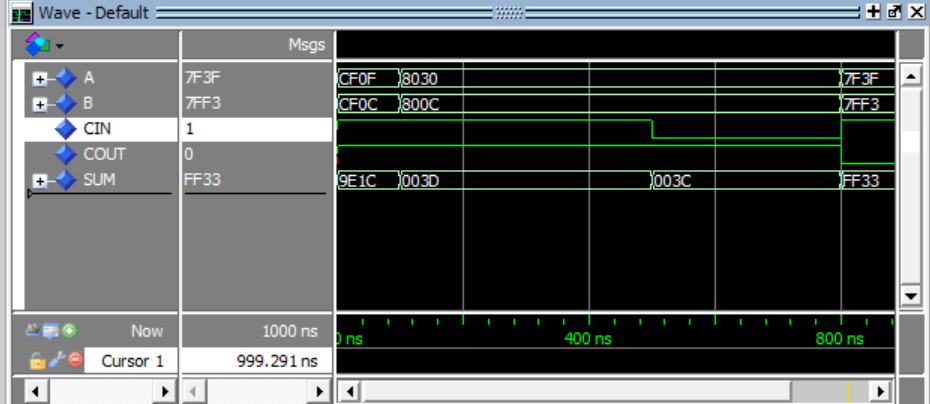
A1 <= "1100111100001111","1000000000110000" AFTER 100 NS, "0111111100111111" AFTER 800 NS;

B1 <= "1100111100001100","1000000000001100" AFTER 100 NS, "0111111111110011" AFTER 800 NS;

CIN1 <= '1','0' AFTER 500 NS, '1' AFTER 800NS;

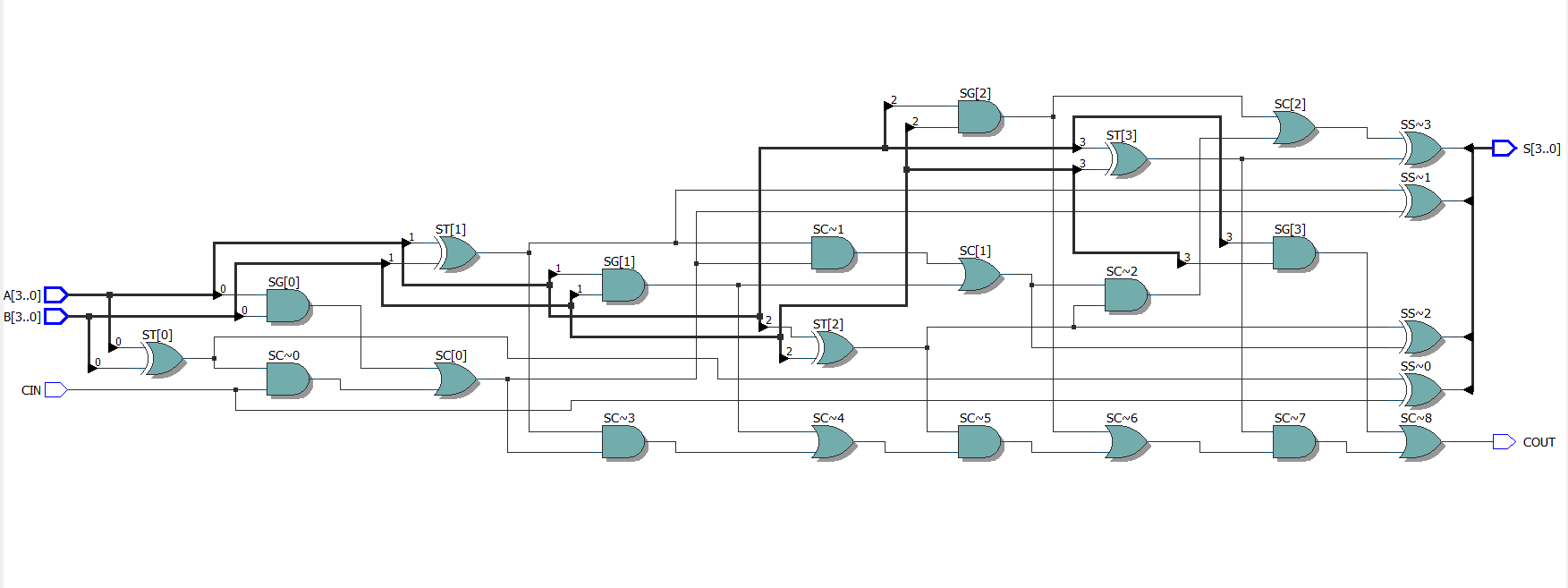
END ADDER16\_C\_arch;

4.仿真波形图

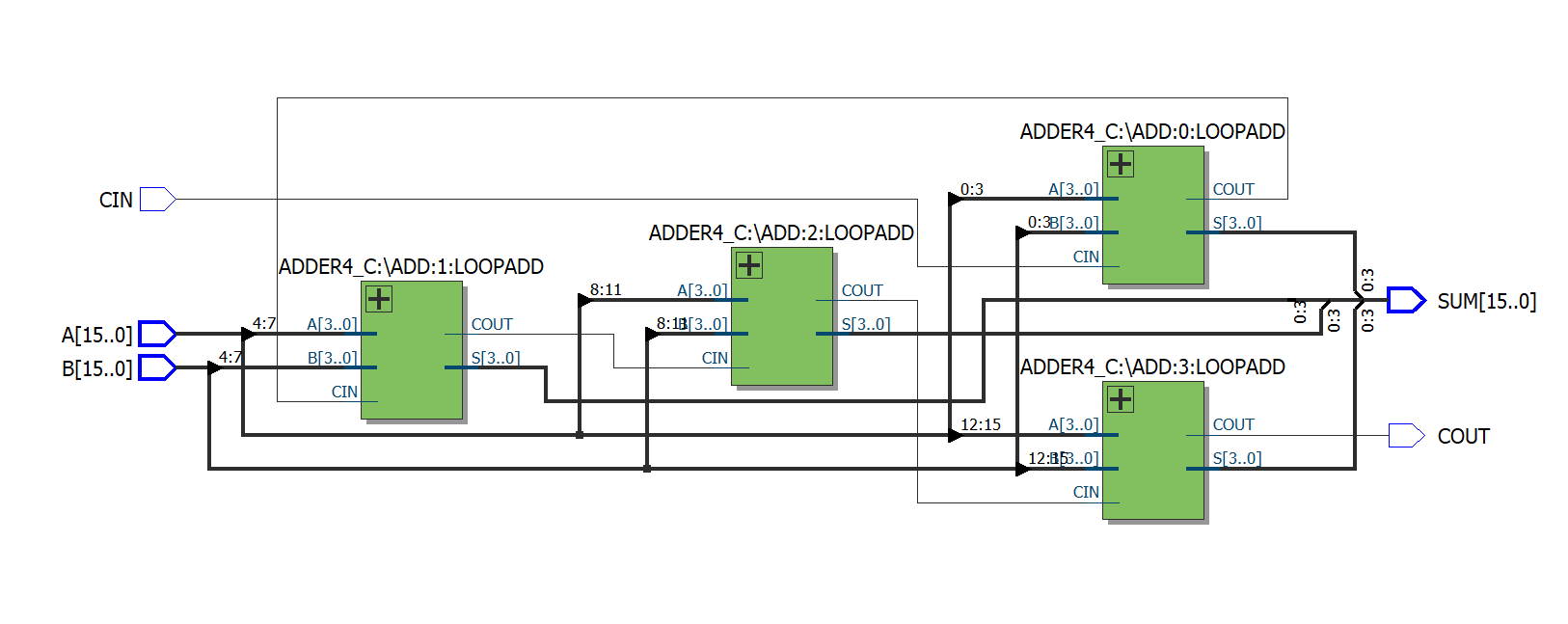


5.门级电路图

（1）四位超前二进制全加器



（2）十六位超前二进制全加器



6.结果分析与思考

限于实现完全的16位全加过于复杂，该设计实现了一定的超前全加功能（四位全加器内部实现了超前进位），而两个四位全加器之间的关系仍然是同步的 。但即便如此依然可以看出，数据相对16全加器级联的方式几乎没有抖动

### 【第四个实验】

1.实验原理

8421BCD码的运算规则与正常的二进制编码有所不同，它的每四位可以看作一个十进制数，满足十进制加法的规则，比如84+40=124而非C4，只需将高四位与第四位拆开，分别按十进制方法相加运算即可

2.设计模块代码（Design Block）

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY BCDADDER8 IS

PORT (A : IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

B : IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

CIN : IN STD\_LOGIC;

SUM : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0);

COUT : OUT STD\_LOGIC);

END ENTITY BCDADDER8;

ARCHITECTURE BHV OF BCDADDER8 IS

BEGIN

PROCESS(A,B,CIN)

VARIABLE DATA: STD\_LOGIC\_VECTOR(9 DOWNTO 0);

BEGIN

DATA :="0000000000";

DATA(4 DOWNTO 0):= ('0'&A(3 DOWNTO 0))+('0'&B(3 DOWNTO 0))+("0000"&CIN);

IF (DATA(4 DOWNTO 0) > 9) THEN

DATA(4) := '1';

END IF;

DATA(9 DOWNTO 5):= ('0'&A(7 DOWNTO 4))+('0'&B(7 DOWNTO 4))+("0000"&DATA(4));

IF (DATA(4 DOWNTO 0) > 9) THEN

DATA(4 DOWNTO 0) := DATA(4 DOWNTO 0) - "01010";

END IF;

IF (DATA(9 DOWNTO 5) > 9) THEN

DATA(9 DOWNTO 5) := DATA(9 DOWNTO 5) - "01010";

DATA(9) := '1';

END IF;

SUM <= DATA(8 DOWNTO 5)&DATA(3 DOWNTO 0);

COUT <= DATA(9);

END PROCESS;

END ARCHITECTURE BHV;

3.激励模块代码（Test Bench）

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

ENTITY BCDADDER8\_vhd\_tst IS

END BCDADDER8\_vhd\_tst;

ARCHITECTURE BCDADDER8\_arch OF BCDADDER8\_vhd\_tst IS

SIGNAL A1 : STD\_LOGIC\_VECTOR(7 DOWNTO 0);

SIGNAL B1 : STD\_LOGIC\_VECTOR(7 DOWNTO 0);

SIGNAL CIN1 : STD\_LOGIC;

SIGNAL COUT1 : STD\_LOGIC;

SIGNAL SUM1 : STD\_LOGIC\_VECTOR(7 DOWNTO 0);

COMPONENT BCDADDER8

PORT (

A : IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

B : IN STD\_LOGIC\_VECTOR(7 DOWNTO 0);

CIN : IN STD\_LOGIC;

COUT : OUT STD\_LOGIC;

SUM : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0)

);

END COMPONENT;

BEGIN

i1 : BCDADDER8

PORT MAP (

A => A1,

B => B1,

CIN => CIN1,

COUT => COUT1,

SUM => SUM1

);

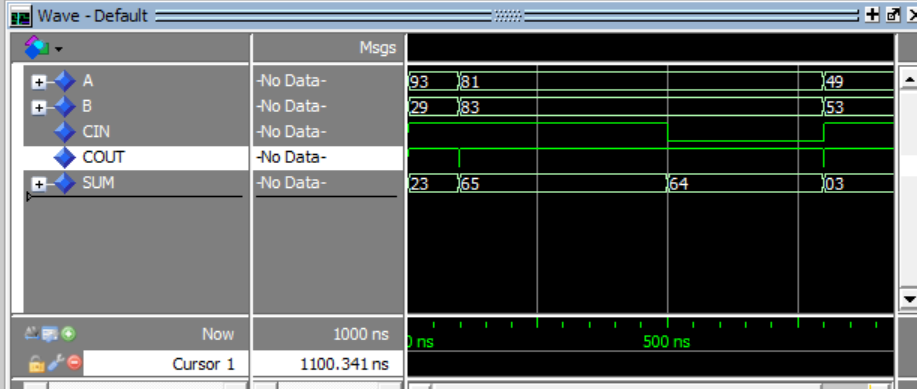
A1 <= "10010011","10000001" AFTER 100 NS, "01001001" AFTER 800 NS;

B1 <= "00101001","10000011" AFTER 100 NS, "01010011" AFTER 800 NS;

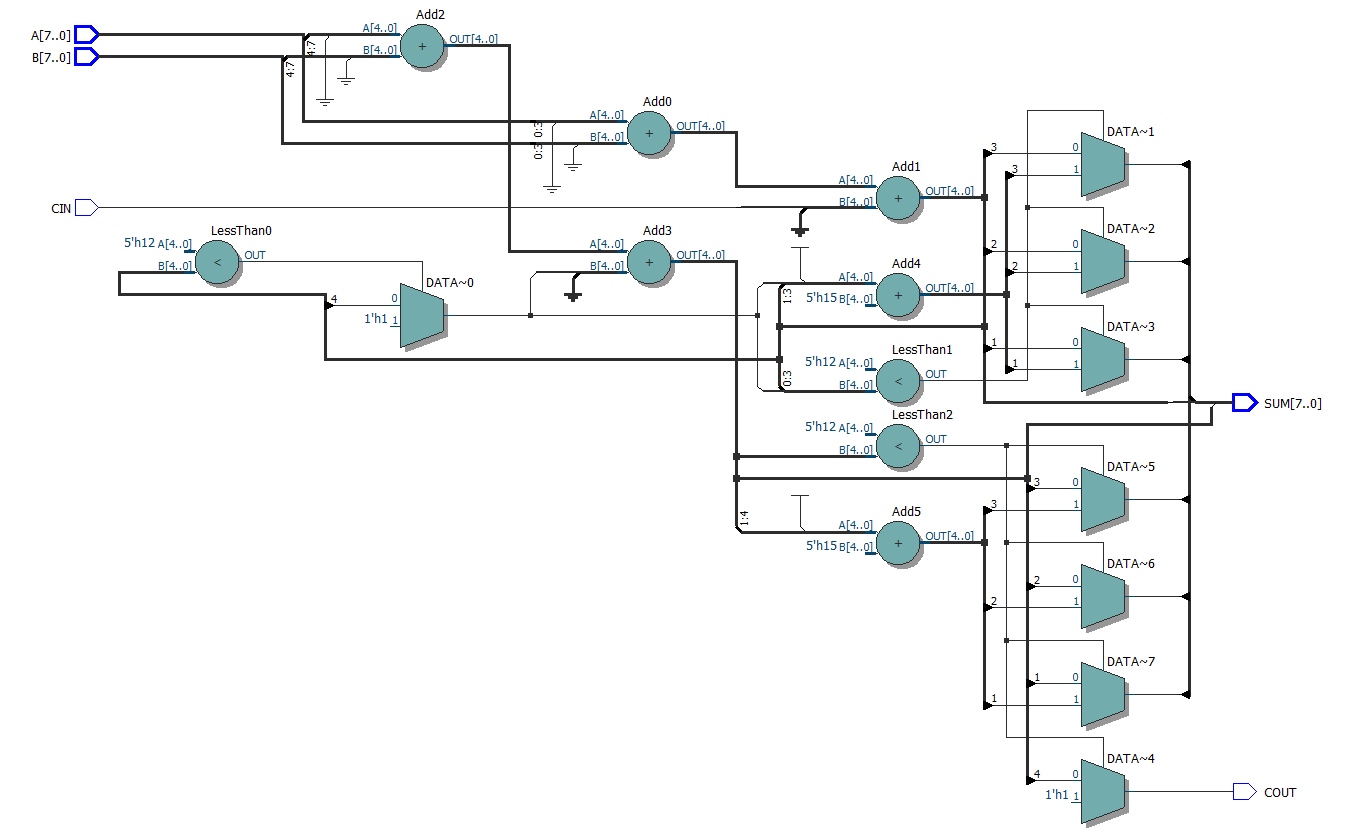
CIN1 <= '1','0' AFTER 500 NS, '1' AFTER 800NS;

END BCDADDER8\_arch;

4.仿真波形图



5.门级电路图



6.结果分析与思考

可见，该设计可以满足要求，缺点在于该方法设置了一个10位数组计算8位数，并非十分巧妙，有继续改进提升的空间，其余的一些方式也是可行的，如先进行完相加后再用除十取余法等转换为BCD码