**实验二 组合电路设计实验报告**

**姓名**：曾泇睷  **学号**：1820221053

**班级**：07812201 **手机**：15101578062

注：黑色字体内容不能改动，蓝色字体内容（为示例或说明）需删除和修改。

1. **实验题目**

设计一个组合电路，输入一个3位的数字，输出一个6位的二进制数字，且输出数字的值等于输入数字值的平方。

1. **实验约束**

* 电路设计时只能使用或非门和非门进行实现。
* 采用Verilog实现时使用结构化描述方式。

1. **电路设计**
   1. **规范化**

输入3位的数字，也就是二进制000到111的数字，对应十进制0-7，而输出位为6位，分别对应输入的数值的平方，结果显然只有0，1，4，9，16，25，36，49八种结果。

* 1. **形式化**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 输入 | | | 输出 | | | | | |
| A | B | C | U | V | W | X | Y | Z |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |

U = Σm(6, 7)

V = Σm(4, 5, 7)

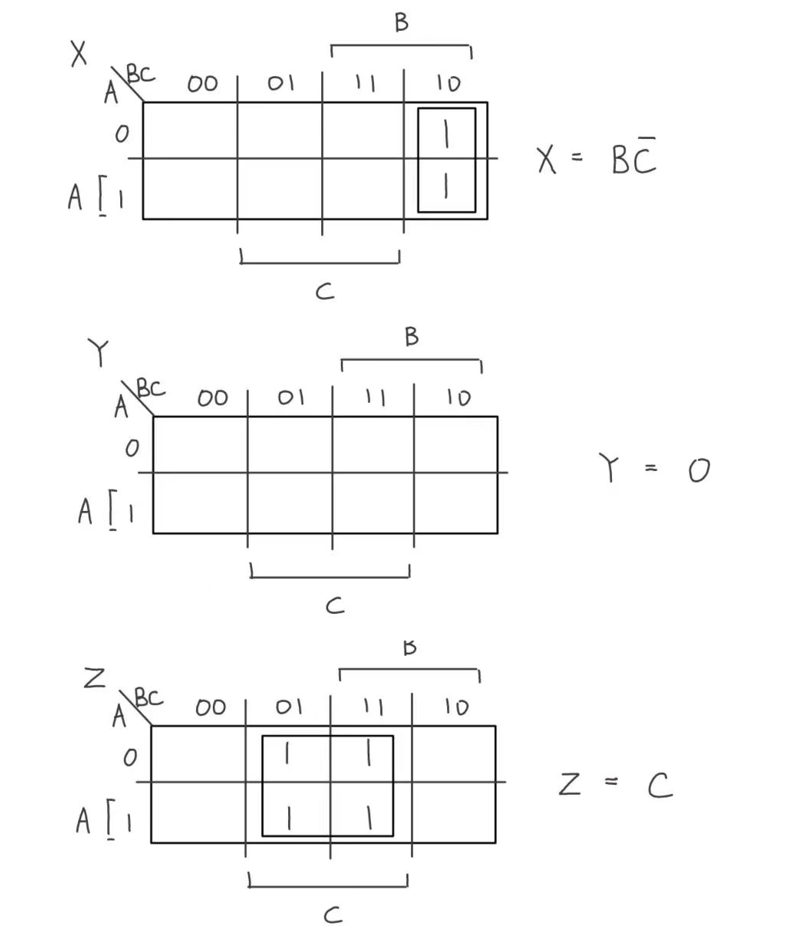
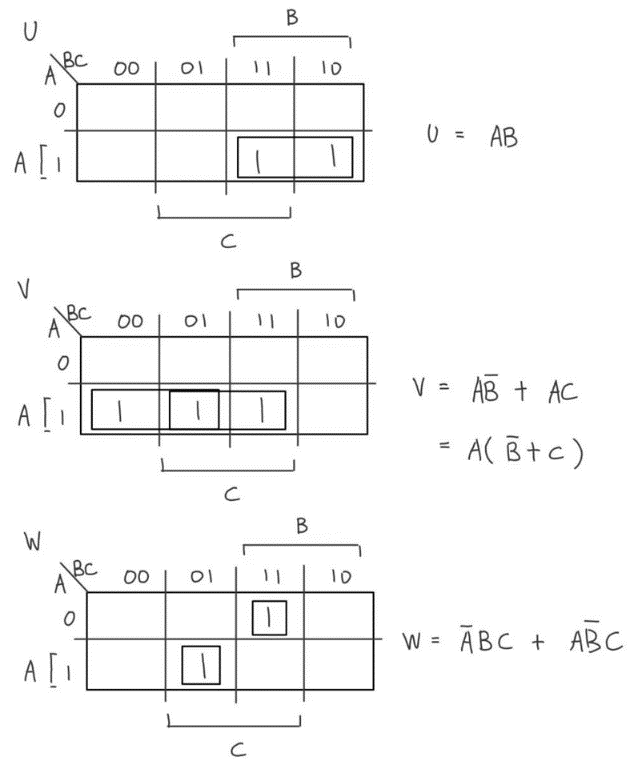
W = Σm(3, 5)

X = Σm(2, 6)

Y = 0

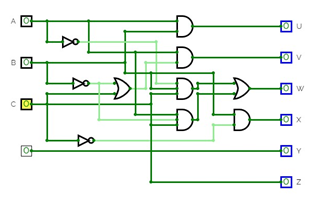
Z = Σm(1, 3, 5, 7)

* 1. **优化**

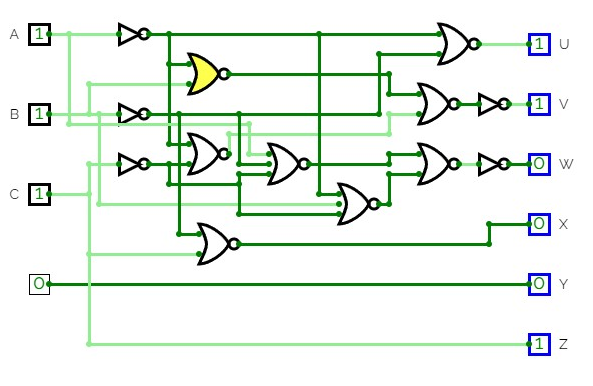


* 1. **工艺映射**

工艺映射前的电路图



工艺映射后的电路图



1. **电路实现**

`timescale 1ns / 1ps

module square(

input a,

input b,

input c,

output[5:0] value

);

wire na, nb, nc; //分别对应非A、非B、非C

wire x, nw, w, anbc, nabc, anb, ac, nv, v, u; //对应X、非W、W、A(非B)C、(非A)BC、A(非B)、AC、非V、V、U

not notA(na, a); //计算非A

not notB(nb, b); //计算非B

not notC(nc, c); //计算非C

nor nor1(x, nb, c); //利用异或计算X

nor nor2(anbc, na, b, nc); //利用异或计算 A(非B)C

nor nor3(nabc, a, nb, nc); //利用异或计算 (非A)BC

nor nor4(nw, anbc, nabc); //利用异或计算非W

not notNW(w, nw); //非门得到W

nor nor5(anb, na, b); //利用异或计算A(非B)

nor nor6(ac, na, nc); //利用异或计算AC

nor nor7(nv, anb, ac); //利用异或计算非V

not notNV(v, nv); //非门得到V

nor nor8(u, na, nb); //异或计算U

// 对输出每一位赋予相应的值

assign value[0] = c;

assign value[1] = 0;

assign value[2] = x;

assign value[3] = w;

assign value[4] = v;

assign value[5] = u;

endmodule

1. **电路验证**
   1. **TestBench**

`timescale 1ns / 1ps

module testbench(

);

reg A, B, C; //声明输入变量

wire[5:0] value; //声明输出变量

//让A变量在20ns后改变为1

initial begin

A = 1'b0;

#20 A = 1'b1;

end

//让B变量每隔10ns就变化一次

initial begin

B = 1'b0;

#10 B = 1'b1;

#10 B = 1'b0;

#10 B = 1'b1;

end

//让C变量每隔5ns就变化一次

initial begin

C = 1'b0;

#5 C = 1'b1;

#5 C = 1'b0;

#5 C = 1'b1;

#5 C = 1'b0;

#5 C = 1'b1;

#5 C = 1'b0;

#5 C = 1'b1;

end

//因此每隔5ns，可得到ABC分别为：000，001，010，011，100，101，110，111

square square(

.a(A),

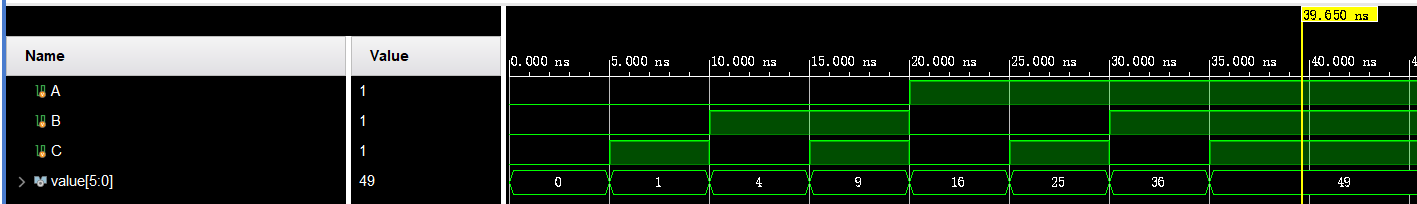
.b(B),

.c(C),

.value(value)

);

endmodule

* 1. **仿真结果**

在testbench中，让输入每5ns改变一次，依次变化为000、001、010、011、100、101、110、111，在仿真结果中可以看到符合预期结果。

而输出value应分别对应输入值的平法，亦是0、1、4、9、16、25、36、49，仿真结果给出的输出结果正确，符合预期结果。

1. **实验心得**

通过本次实验，我深入理解了组合逻辑电路的设计原理。尤其是在使用基本逻辑门（或非门和非门）的情况下，如何通过组合这些基本逻辑门实现复杂的逻辑功能，在实验中实现了输入3位数字的平方运算。

我对Verilog的结构化描述方式有了更深刻的理解和实际操作经验。结构化描述方式不仅使代码更具有可读性和可维护性，同时也提高了代码的模块化程度。