**实验报告**

实验题目： 数制系统 日期： 2018 年 10 月 19 日

姓名: 罗晏宸 学号: PB17000297 成绩:\_\_\_\_\_\_\_\_\_\_\_\_

**实验目的：**

1. 学习使用Verilog语言定义不同进制的数
2. 设计能将数据从一个进制转化到另一个的组合电路（以二进制至十进制为例）并加以实现
3. 设计能实现简单的加法操作的组合电路并加以实现
4. 学习超前进位加法电路的原理并设计实现以提高加法操作速度

**实验内容**（截图、照片与代码）

**Lab2\_1\_1**

代码Lab2\_1\_1.v

`timescale 1ns / 1ps

module lab2\_1\_1(

output [7:0] an,

output [6:0] seg

);

reg [3:0] x = 4'd7;

assign seg[0] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && ~x[1] && ~x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[1] = (~x[3] && x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && x[1] && ~x[0]) | (x[3] && (x[2] | x[1]));

assign seg[2] = (~x[3] && ~x[2] && x[1] && ~x[0]) | (x[3] && (x[2] | x[1]));

assign seg[3] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && ~x[1] && ~x[0]) | (~x[3] && x[2] && x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[4] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && ~x[2] && x[1] && x[0]) | (~x[3] && x[2] && ~x[1] && ~x[0]) | (~x[3] && x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && x[1] && x[0]) | (x[3] && ~x[2] && ~x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[5] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && ~x[2] && x[1] && ~x[0]) | (~x[3] && ~x[2] && x[1] && x[0]) | (~x[3] && x[2] && x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[6] = ((~x[3] && ~x[2] && ~x[1] && ~x[0]) | (~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign an[0] = 0;

assign an[1] = 1;

assign an[2] = 1;

assign an[3] = 1;

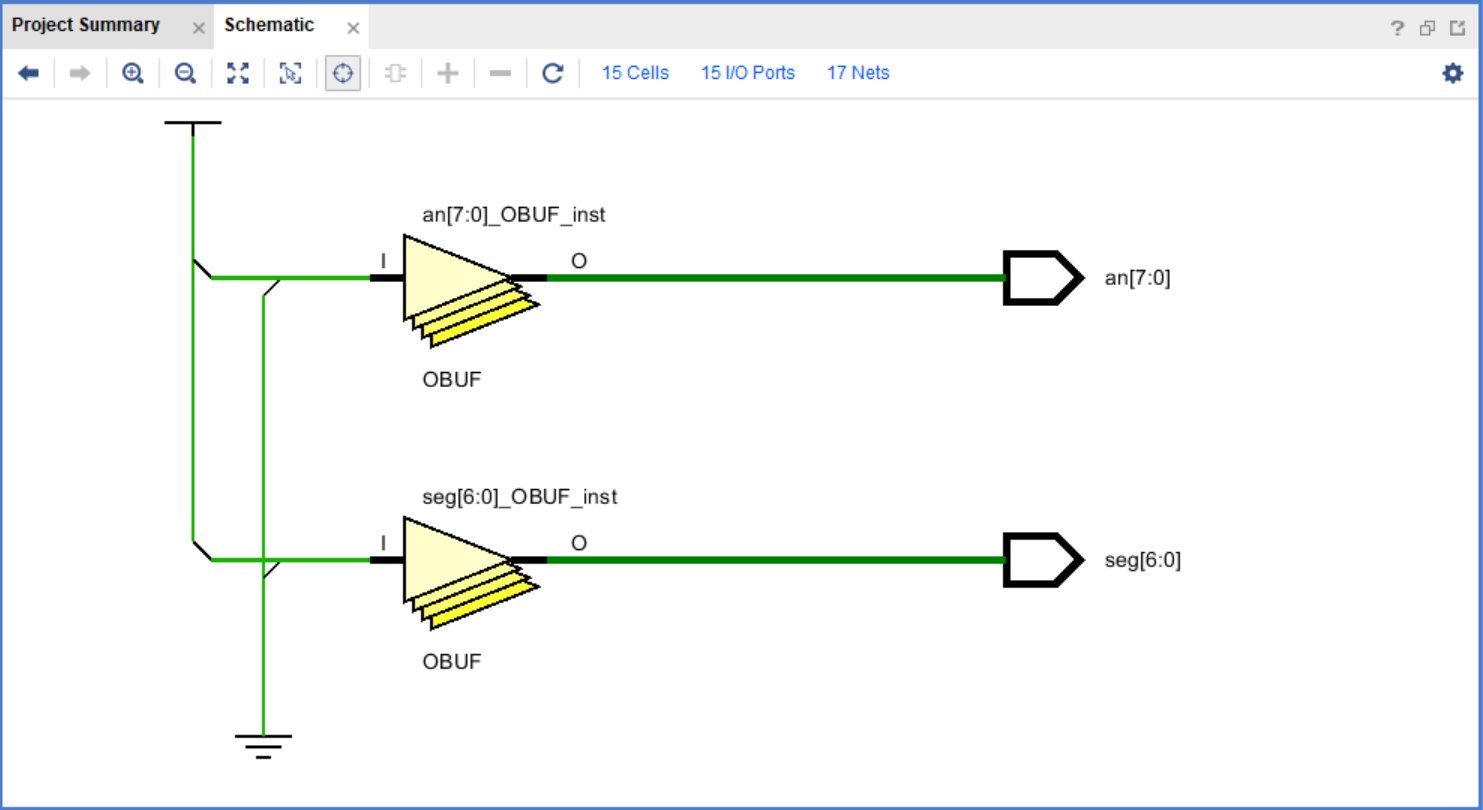
assign an[4] = 1;

assign an[5] = 1;

assign an[6] = 1;

assign an[7] = 1;

endmodule



截图 1-lab2\_1\_1原理图



照片 1-lab2\_1\_1下载

“照片 1-lab2\_1\_1下载”的说明：本工程使用数据流风格实现一位已定义的十进制数在7段数码管上的显示；代码中已定义x = 4'd7,右侧七段数码管0位CA, CB, CC管亮，其余管灭，其余位七段数码管全灭，灯亮的数码管笔画构成数字‘7’字样形成输出。

**Lab2\_2\_1**

代码lab2\_2\_1.v

`timescale 1ns / 1ps

module lab2\_2\_1(

input [3:0] v,

output z,

output [7:0] an,

output [6:0] seg

);

wire [3:0] m;

lab2\_2\_1\_partA (v,z,m);

bcdto7segment\_dataflow (m,an,seg);

endmodule

代码 lab2\_2\_1\_partA.v

`timescale 1ns / 1ps

module lab2\_2\_1\_partA(

input [3:0] v,

output z,

output [3:0] m

);

wire [3:0] CircuitAOutput;

assign z = (v[3] & v[2]) | (v[3] & v[1]);

assign CircuitAOutput[3] = 0;

assign CircuitAOutput[2] = v[1] & v[2];

assign CircuitAOutput[1] = ~v[1];

assign CircuitAOutput[0] = v[0];

assign m[3] = (~z & v[3]) | (z & CircuitAOutput[3]);

assign m[2] = (~z & v[2]) | (z & CircuitAOutput[2]);

assign m[1] = (~z & v[1]) | (z & CircuitAOutput[1]);

assign m[0] = (~z & v[0]) | (z & CircuitAOutput[0]);

endmodule

代码bcdto7segment\_dataflow.v

`timescale 1ns / 1ps

module bcdto7segment\_dataflow(

input [3:0] x,

output [7:0] an,

output [6:0] seg

);

assign seg[0] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && ~x[1] && ~x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[1] = (~x[3] && x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && x[1] && ~x[0]) | (x[3] && (x[2] | x[1]));

assign seg[2] = (~x[3] && ~x[2] && x[1] && ~x[0]) | (x[3] && (x[2] | x[1]));

assign seg[3] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && ~x[1] && ~x[0]) | (~x[3] && x[2] && x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[4] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && ~x[2] && x[1] && x[0]) | (~x[3] && x[2] && ~x[1] && ~x[0]) | (~x[3] && x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && x[1] && x[0]) | (x[3] && ~x[2] && ~x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[5] = ((~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && ~x[2] && x[1] && ~x[0]) | (~x[3] && ~x[2] && x[1] && x[0]) | (~x[3] && x[2] && x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign seg[6] = ((~x[3] && ~x[2] && ~x[1] && ~x[0]) | (~x[3] && ~x[2] && ~x[1] && x[0]) | (~x[3] && x[2] && x[1] && x[0])) && ~(x[3] && (x[2] | x[1]));

assign an[0] = 0;

assign an[1] = 1;

assign an[2] = 1;

assign an[3] = 1;

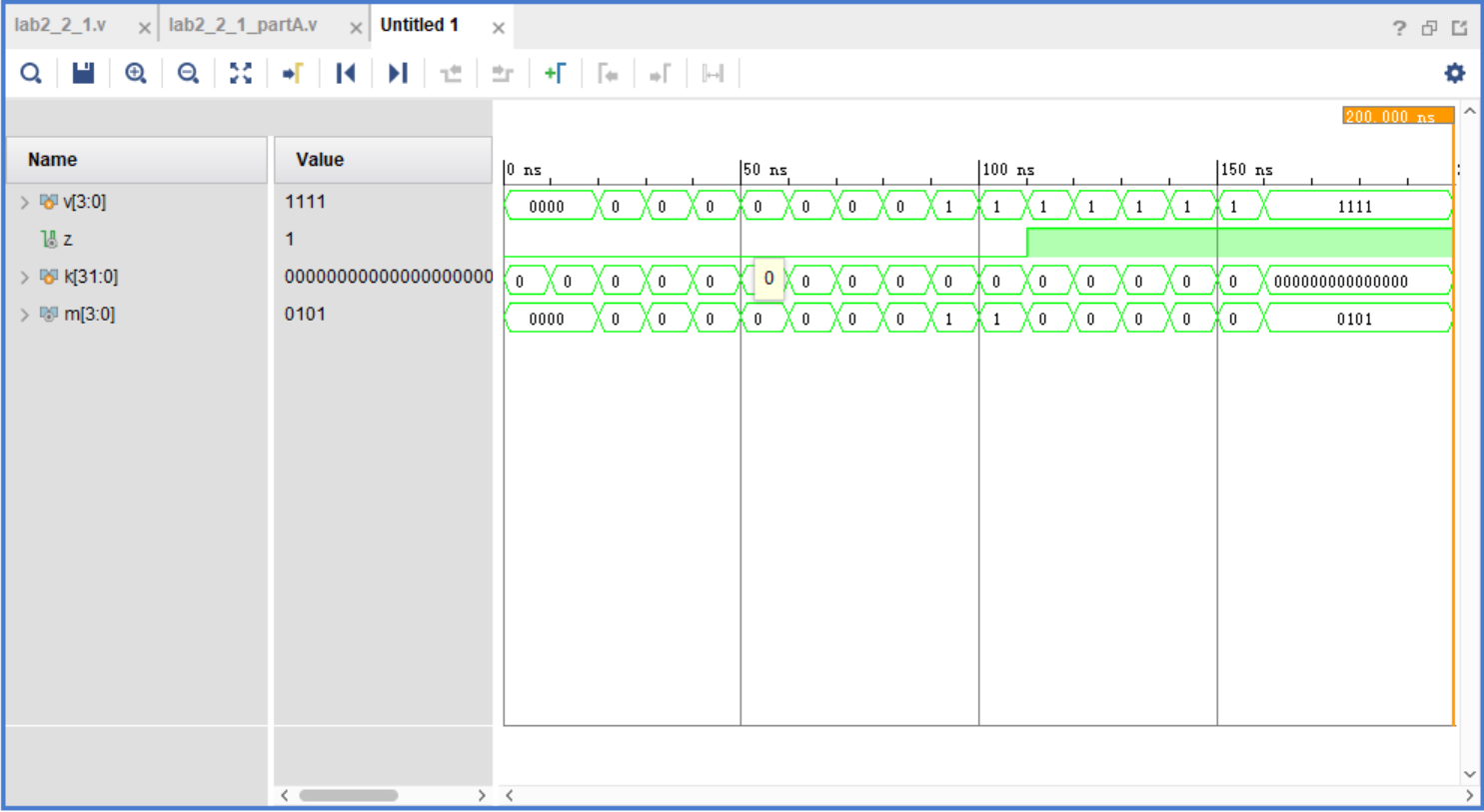
assign an[4] = 1;

assign an[5] = 1;

assign an[6] = 1;

assign an[7] = 1;

endmodule



截图 2-lab2\_2\_1仿真



照片 2-lab2\_2\_1下载A

“照片 2-lab2\_2\_1下载A”的说明：本工程使用数据流风格实现将 4 比特的二进制输入转化为相应的 2 位十进制数（BCD码）后输出到 LED 灯（最高位）和最右边的七段数码管（低位）上；图中开关switch[1]拨至开，其余开关拨至关，表示输入为v=(0010)B；led灯均灭；右侧七段数码管0位CA, CB, CD, CE，CG管亮，其余管灭，其余位七段数码管全灭，灯亮的数码管笔画构成数字‘2’字样形成输出，表示输出为(2)D.



照片 3-lab2\_2\_1下载B

“照片 3-lab2\_2\_1下载B”的说明：本工程使用数据流风格实现将 4 比特的二进制输入转化为相应的 2 位十进制数（BCD码）后输出到 LED 灯（最高位）和最右边的七段数码管（低位）上；图中开关switch[0]，switch[1]，switch[3]拨至开，其余开关拨至关，表示输入为v=(1011)B；灯led[0]亮，其余灯灭；右侧七段数码管0位CB, CC管亮，其余管灭，其余位七段数码管全灭，灯亮的数码管笔画构成数字‘1’字样形成输出，表示输出为(11)D.

**Lab2\_3\_1**

代码lab2\_3\_1.v

`timescale 1ns / 1ps

module lab2\_3\_1(

input [3:0] a,

input [3:0] b,

input cin,

output cout,

output [3:0] s

);

wire out[2:0];

fulladder\_dataflow (a[0],b[0],cin,out[0],s[0]);

fulladder\_dataflow (a[1],b[1],out[0],out[1],s[1]);

fulladder\_dataflow (a[2],b[2],out[1],out[2],s[2]);

fulladder\_dataflow (a[3],b[3],out[2],cout,s[3]);

endmodule

代码fulladder\_dataflow.v

`timescale 1ns / 1ps

module fulladder\_dataflow(

input a,

input b,

input cin,

output cout,

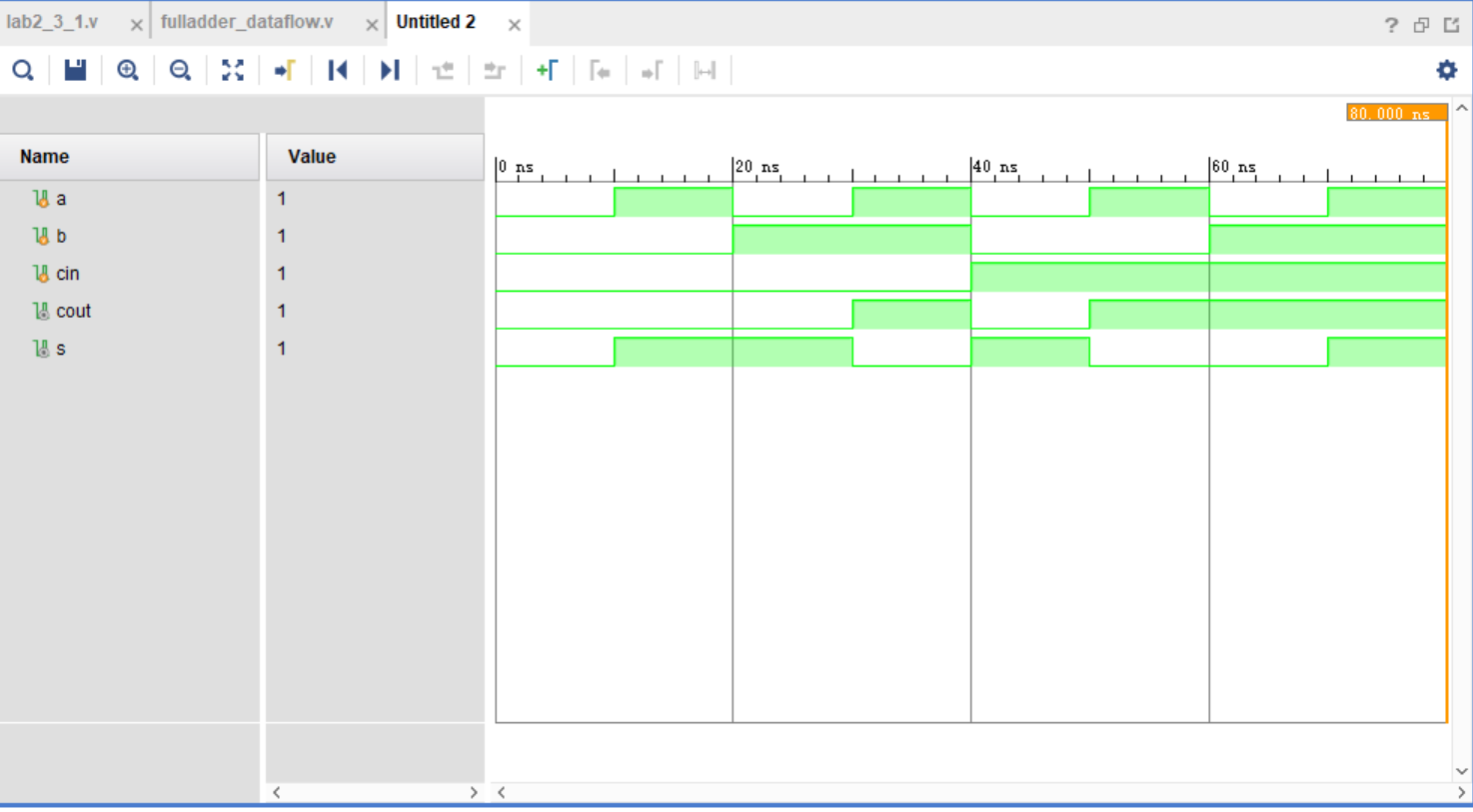
output s

);

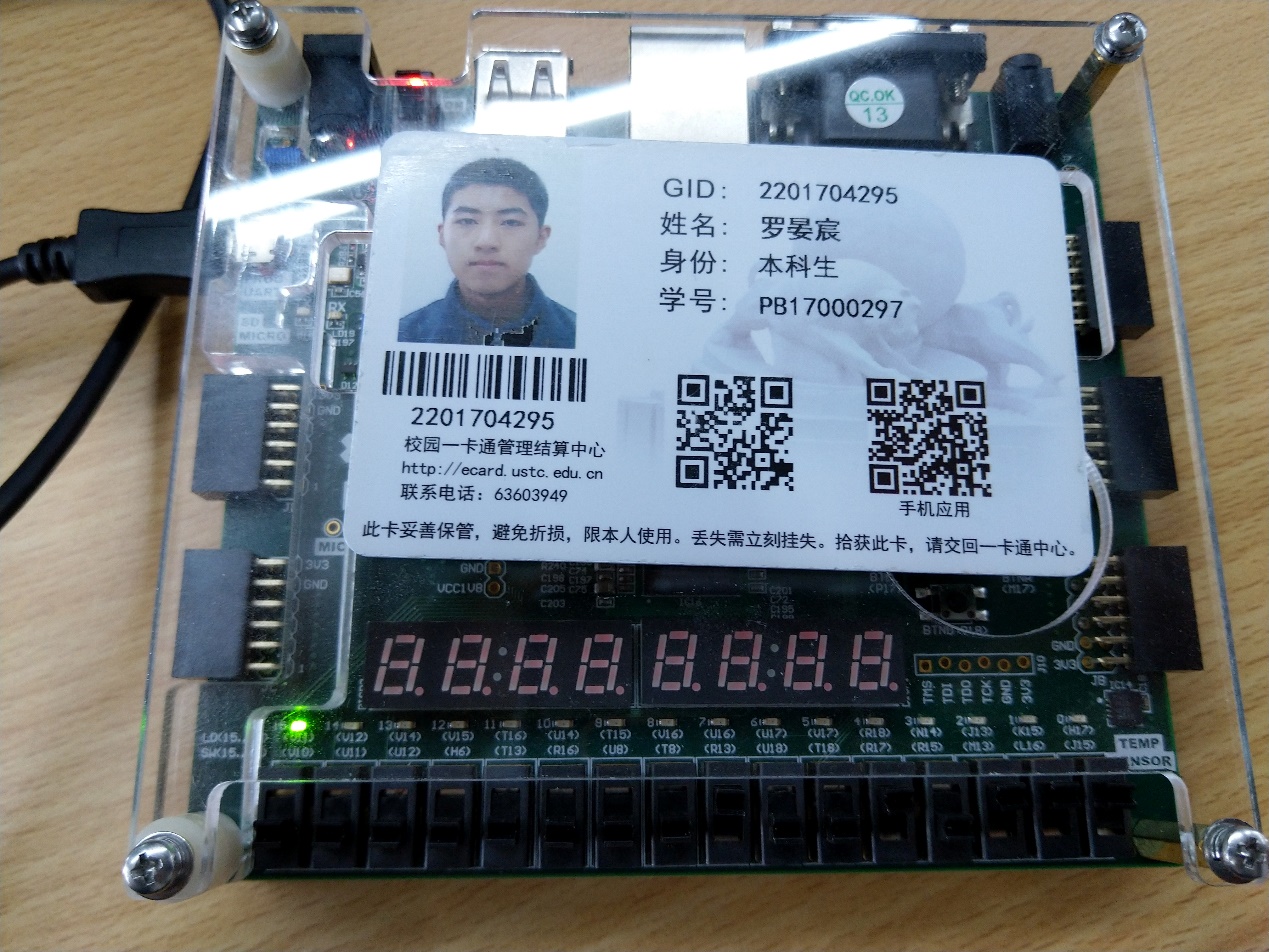
assign s = a ^ b ^ cin;

assign cout = ( a & cin) | (b & cin) | (a & b);

endmodule



截图 3-lab2\_3\_1仿真



照片 4-lab2\_3\_1下载

“照片 4-lab2\_3\_1下载”的说明：本工程使用数据流风格实现4 比特波纹进位加法器；图中开关switch[0]，switch[1]，switch[2]，switch[4]，switch[7]，拨至开，其余开关拨至关，表示输入为a=(0111)B, b=(1001)B，cin=0；led[15]亮，其余灯灭，表示输出为cout=1,s=(0000)B.

**实验总结：**

在本实验中，我学习了如何定义不同进制的数，并使用数据流设计不同的转换进制或数码的电路。并利用这些知识实现了4位加法器，练习了实验操作与Verilog语法知识。