同济大学计算机系

数字逻辑课程综合实验报告

学　　号 _____/ / /_____

姓　　名 _____/ / /_____

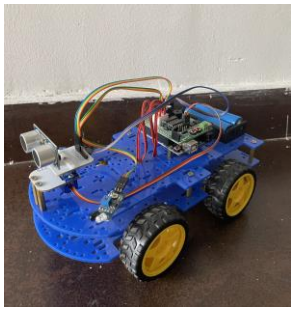
专　　业 _____计算机科学与技术_____

授课老师 _____/ / /_____

目录

一、 实验内容

1.项目内容概述



SLAM (simultaneous localization and mapping),也称为 CML (Concurrent Mapping and Localization), 即时定位与地图构建，或并发建图与定位。问题可以描述为：将一个机器人放入未知环境中的未知位置，是否有办法让机器人一边逐步描绘出此环境完全的地图，同时一边决定机器人应该往哪个方向行进。例如扫地机器人就是一个很典型的 SLAM 问题，所谓完全的地图（a consistent map）是指不受障碍行进到房间可进入的每个角落。

受此 AI 智能技术的启发，本项目也试图实现一个类似具有实时位置定位和周围环境感知的系统。

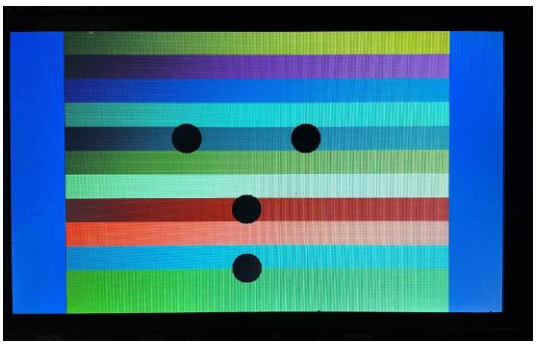
本项目使用 FPGA、VGA 显示器、主从蓝牙模块、L3G4200D 三轴传感器（后改为板子旋钮）、arduino 单片机、超声波雷达、红外霍尔传感器和部分智能小车基础配件（如步进电机等）。实现了简易版的 SLAM 智能车。因此本项目智能车命令为 slep（即 steerable localization environmental perception）--可操纵的实时定位环境感知智能车。

将 slep 放入陌生环境中，可以对小车进行方向操纵，小车对周围环境进行检测，将自身位置信息及周围环境距离信息反映在显示器上，以实现定位与地图构建的初步功能。

2.界面设置

slep 位置信息将实时反馈到 VGA 进行显示，这里针对 VGA 界面信息进行介绍。

初始页面：

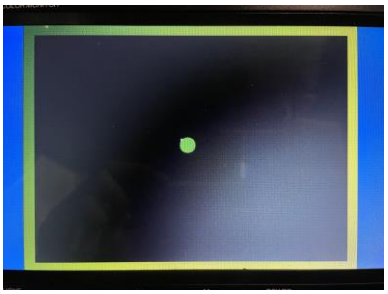


该图形为 slep 的标志，形似一个向前行走的叉，寓意一直向前，也暗含 slep 用于实时定位与环境感知的功能内涵。

实时定位界面：

图中绿点会根据小车的移动实时变化位置

遇到障碍界面：



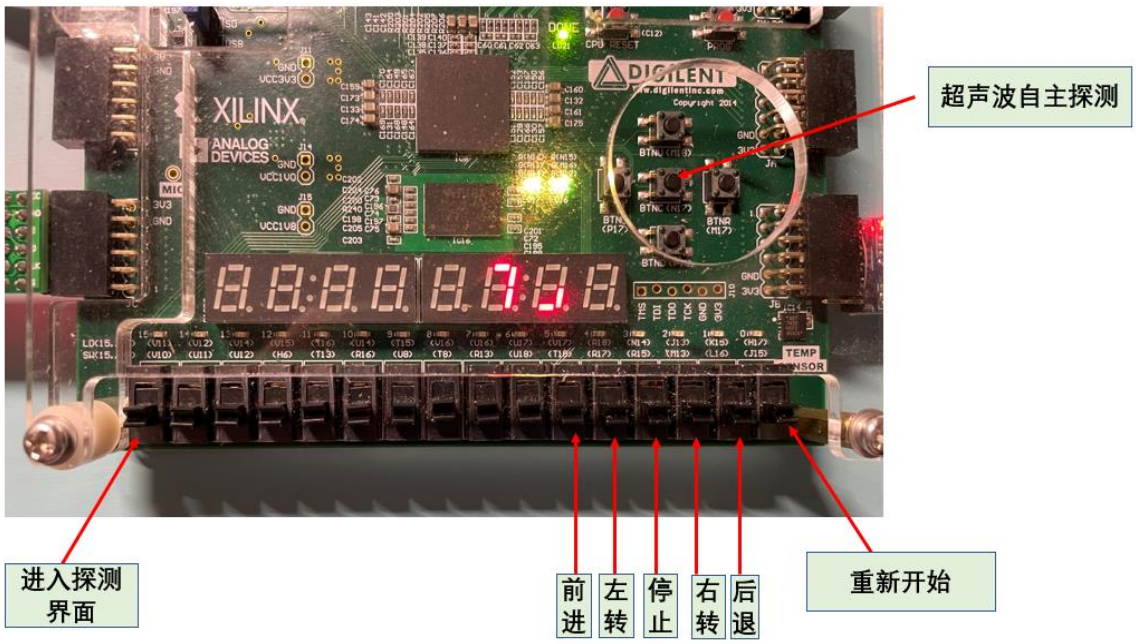
小车探测到障碍，蜂鸣器将启动，小车无法前进，VGA 界面将出现紫色的障碍物标志。此时，只有执行后退，小车才能接触此状态。

多次遇障的界面：

除自行探测障碍外，小车可根据指令，转动超声波探头，自行探测障碍，并在界面中显示最近的障碍物位置。图片中紫色点为小车探测到的障碍物信息。

3.操作说明

受限于三轴传感器的灵敏度问题，因此采用 FPGA 上按钮实现小车的远程操控。



4.器件介绍

1.Nexys 4 DDR Artix-7

由 Xilinx 公司开发出的一款现场可编程门阵列（FPGA）开发板。

2.VGA

VGA(Video Graphics Array)是 IBM 在 1987 年随 PS/2 机一起推出的一种 视频传输标准，具有分辨率高、显示速率快、颜色丰富等优点。

3.HC-06 和 HC-05 蓝牙

HC-06 和 HC-05 的硬件相同，都采用 CSR (Cambridge Silicon Radio) 公司的 BC417143 芯片，支持蓝牙 2.1+EDR 规范，只是芯片内部的控制程序不同。在 FPGA 上使用 HC-6 模块作为主机发起连接信息，Arduino 单片机上使用 HC-5 模块作为从机接受信息。

4.L3G4200D

三轴数字输出陀螺仪——由意法半导体（ST）集团制造的一款三轴数 字输出陀螺仪。用于测量空间三个维度的角速度，同时支持 SPI 和 I 2C 协议。在本次实验中尝试使用的是 SPI 协议。

5.arduino 单片机

微控制单元(Microcontroller Unit, 即 MCU) ，又称单片微型计算机(Single Chip Microcomputer).是一种集成电路芯片， 将 CPU、存储器、I/O 接口等装配在一块芯片上，就构成了一台单片微型计算机（简称单片机）。Arduino 就是一台单片机，由 arm 公司开发。

6.舵机

HG14-M 舵机，由北京汉库公司生产，采用传统的 PWM 协议，优点是已经产业化，成本低，旋转角度大（目前所生产的都可达到 185 度）。

7.超声波雷达 HC-SR04

HC-SR04 超声波测距模块可提供 2cm-400cm 的非接触式距离感测功能， 测距精度可达高到 3mm；模块包括超声波发射器、接收器与控制电路。采用 IO 口 TRIG 触发测距，给最少 10us 的高电平信呈。模块自动发送 8 个 40khz 的方波，自动检测是否有信号返回；

有信号返回，通过 IO 口 ECHO 输出一个高电平，高电平持续的时间就是超波从发射到返回的时间。测试距离=(高电平时间*声速(340M/S))/2。

8.红外霍尔传感器

该霍尔传感器有四个引脚，“vcc”接在单片机的“+5v”引脚（即单片机输出一个五伏的电压）“GND”对应单片机的“GND”（负极），“D0”对应单片机的“D2-D12”引脚（回数字模拟量），同理“A0”对应单片机的“A0-A7”引脚

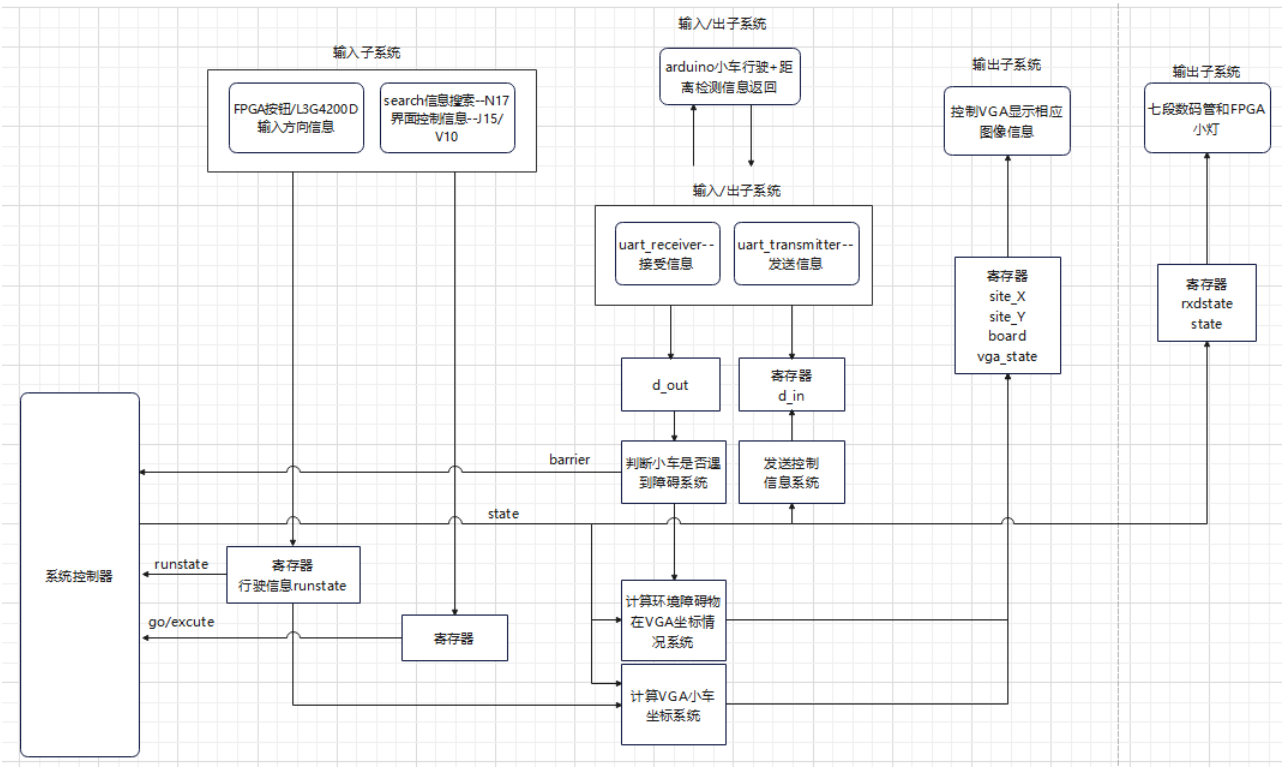
这个类型的霍尔传感器可以测试磁场，电流大小。进而通过红外接收进行测距。

9.小车配件

如：蜂鸣器、步进电机等通用配件，不作详细介绍。

二、 SLEP 数字系统总框图

1.FPGA 系统总框图：



输入子系统：通过输入，对小车行驶状态；是否超声波扫描进行控制。

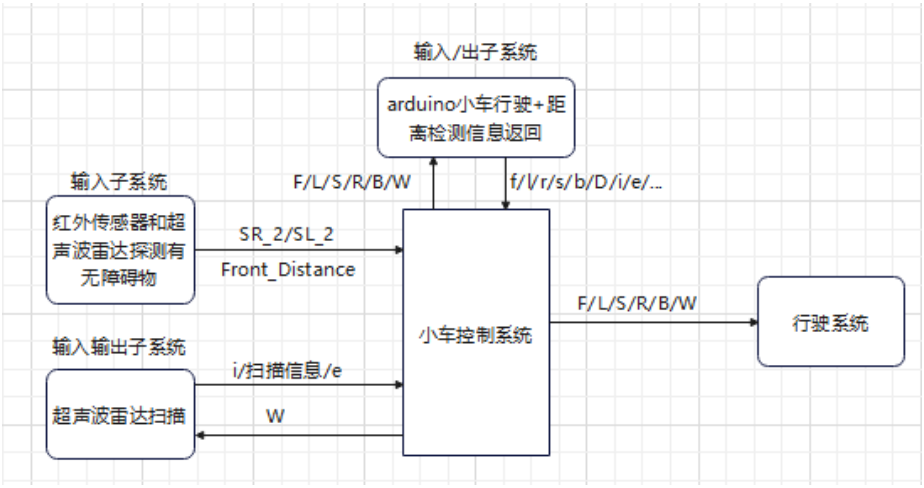
Uart 输入/输出子系统：控制蓝牙 uart 串口，与小车进行通讯。

VGA 输出子系统：通过状态信息和 broad 和 xy 坐标信息，将小车位置和周围障碍物情况在 VGA 上进行显示。

七段数码管和小灯输出子系统：输出小车所处状态情况和 uart 接收到的信号信息。

Arduino 小车输入/输出子系统：对环境进行检测，控制小车行驶，通过 uart 与 FPGA 进行通讯。

2.Arduino 小车系统结构：



小车控制系统：根据 uart 接收信息对小车当前状态进行控制。

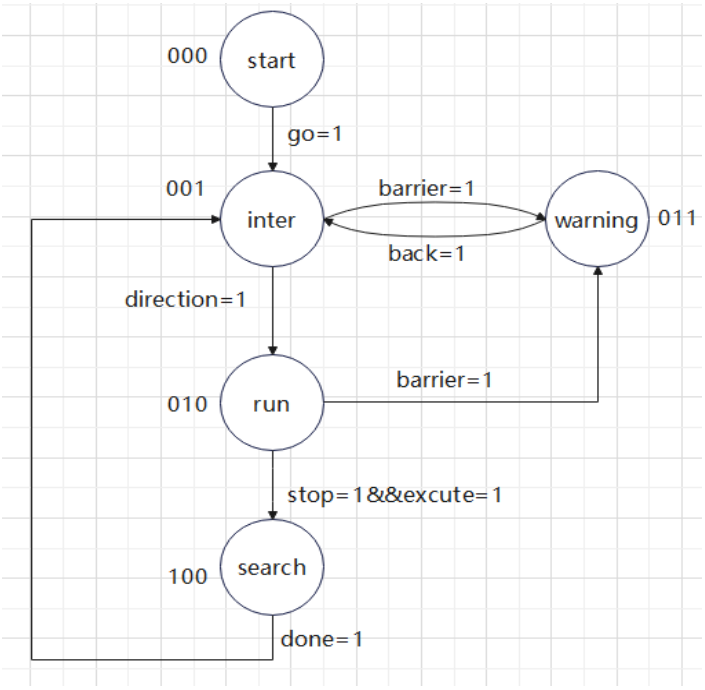
输入子系统：一直对周围环境进行检测，若测得障碍物较近，立即返回小车控制系统。

输入/输出系统：接收到 W 对周围环境进行检测，返回扫描信息。

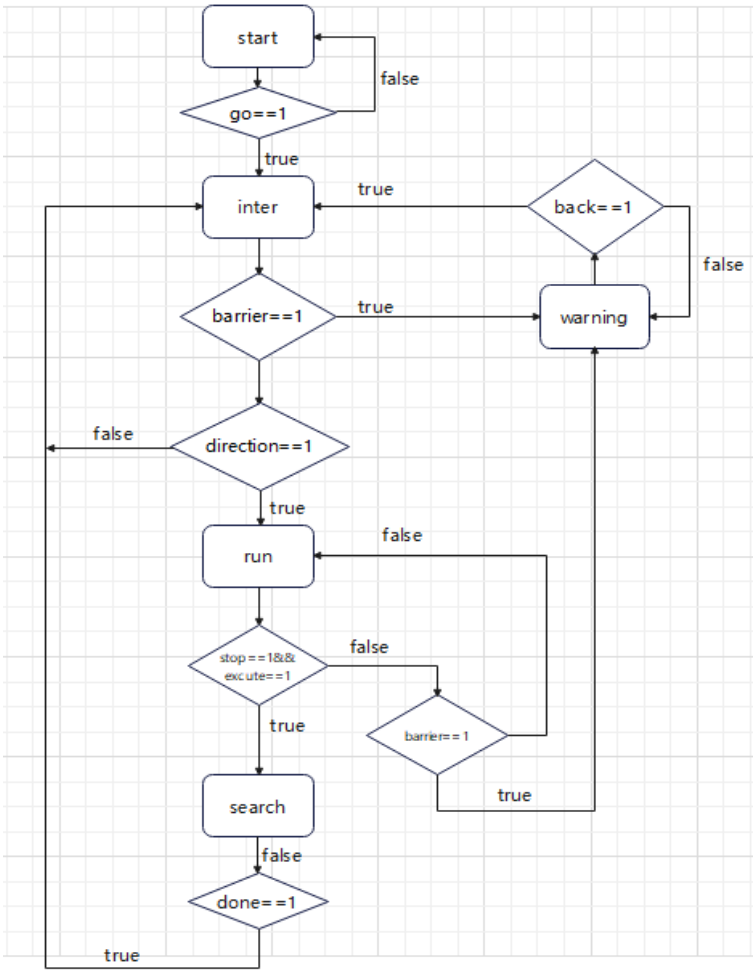
行驶系统：控制小车行驶。

三、 系统控制器设计

1..状态转移图



2.ASM 流程图



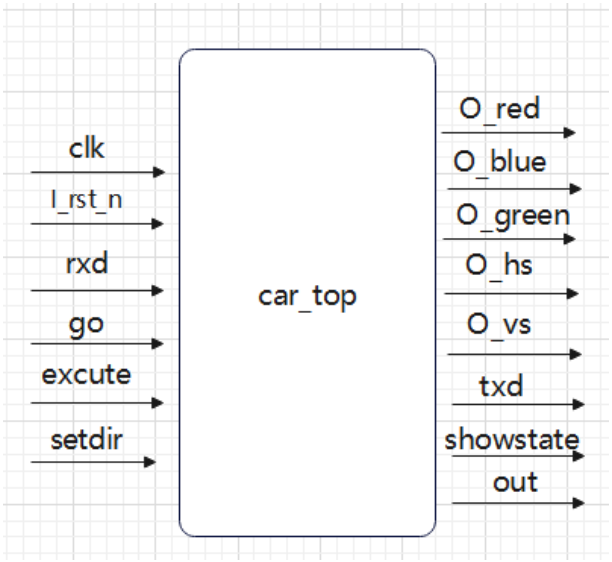
3.状态转移真值表

PS（现态）										NS（次态）		
C	B	A	go	back	direction	barrier	stop	excute	done	C(D)	B(D)	A(D)
0	0	0	1	X	X	X	X	X	X	0	0	1
0	0	1	X	X	X	1	X	X	X	0	1	1
0	0	1	X	X	1	0	X	X	X	0	1	0
0	1	1	X	1	X	X	X	X	X	0	0	1
0	1	0	X	X	X	X	1	1	X	1	0	0
0	1	0	X	X	X	1	1/0	1/0	X	0	1	1
1	0	0	X	X	X	X	X	X	1	0	0	1

四、 子系统模块建模

1.car_top 顶层模块

slep 的顶层模块，内含系统控制器，以及连接各个模块。




```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:
// Design Name:
// Module Name: car_top
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
```

```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

module car_top(
    input                clk      ,
    input                I_rst_n  ,   // low level effective
    input                rxd      ,
    input                go       ,   // high level effective
    input                excute   ,   // high level effective
    input          [4:0] setdir   ,   // for test traxial...
    output   wire        txd      ,
    output   wire  [3:0] O_red    ,   // VGA's red   channel
    output   wire  [3:0] O_green  ,   // VGA's green channel
    output   wire  [3:0] O_blue   ,   // VGA's blue  channel
    output               O_hs     ,   // VGA
    output               O_vs     ,   // VGA
    output   reg   [4:0] showstate,   // show for test
    output   wire  [6:0] out
    );
/*************************************************************/
// inter signals
/*************************************************************/
//uart
reg rdn;                  //low level effective
reg wrn;                  //low level effective
reg [7:0] d_in;
wire [7:0] d_out;
wire r_ready;
wire t_empty;
//vga
reg signed [15:0] site_X; //0-600
reg signed [15:0] site_Y; //0-440
reg [1485:0] broad;
reg [4:0] vga_state;
reg [9:0] angle;          //don't define in vga.v
//triaxial
reg running;
//display7
reg [3:0]playseven;

//state defination
parameter NONE     =  5'b00000 ;
parameter START    =  5'b00001 ;
```

```
parameter INTER    =  5'b00010 ;
parameter RUN      =  5'b00100 ;
parameter WARNING  =  5'b01000 ;
parameter SEARCH   =  5'b10000 ;
reg [4:0]state;

//others
reg barrier;
reg goback;
reg stop;
reg direction;
reg done;
wire [4:0]runstate;//F10000 L01000 S00100 R00010 B00001
parameter Forward = 5'b10000 ;
parameter Left    = 5'b01000 ;
parameter Stop    = 5'b00100 ;
parameter Right   = 5'b00010 ;
parameter Back    = 5'b00001 ;

//parameter Watch   = 5'b11111 ;

reg [7:0]rxdstate;//D10000000 i01000000 e00100000 s00010000
                  //f00001000 l00000100 r00000010 b00000001
parameter Dann = 8'b10000000 ;
parameter Inii = 8'b01000000 ;
parameter Endd = 8'b00100000 ;
parameter Stoo = 8'b00010000 ;
parameter Forr = 8'b00001000 ;
parameter Leff = 8'b00000100 ;
parameter Rigg = 8'b00000010 ;
parameter Bacc = 8'b00000001 ;

/*************************************************************/
// set module
/*************************************************************/
uart ua(
    .clk(clk)          ,   //100Mhz
    .I_rst_n(I_rst_n)  ,   //low level effective
    .rdn(rdn)          ,   //low level effective
    .wrn(wrn)          ,   //low level effectives
    .rxd(rxd)          ,
    .d_in(d_in)        ,
    .d_out(d_out)      ,
    .r_ready(r_ready)  ,
```

```
    .t_empty(t_empty)  ,
    .txd(txd)
);
vga vg(
    .site_X(site_X)    ,
    .site_Y(site_Y)    ,
    .state(vga_state)  ,   // start-inter-run-search-warning--high level effective
    .broad(broad)      ,   // 60*44
    .I_clk(clk)        ,   // 100Mhz-clock
    .I_rst_n(I_rst_n)  ,   // low level effective
    .O_red(O_red)      ,   // VGA's red   channel
    .O_green(O_green)  ,   // VGA's green channel
    .O_blue(O_blue)    ,   // VGA's blue  channel
    .O_hs(O_hs)        ,
    .O_vs(O_vs)
);
triaxial tr(
    .clk(clk)          ,
    .I_rst_n(I_rst_n)  ,   // low level effective
    .set(setdir)       ,   // will change that...
    .running(running)  ,   // high level effective
    .runstate(runstate)
);
//display7
display7 dis(
    .iData(playseven),.oData(out)
);
/*************************************************************/
//state machine
/*************************************************************/
always @ (posedge clk)
begin
    if(!I_rst_n)begin
        state<=5'b00000;
        //...something need input...
    end
    else begin
        case(state)
            NONE:begin
                state<=START;
            end
            START:begin
                if(go==1'b1)
                    state<=INTER;
```

```
                else
                    state<=START;
            end
            INTER:begin
                if(barrier==1'b1)
                    state<=WARNING;
                else if(direction==1'b1)
                    state<=RUN;
                else
                    state<=INTER;
            end
            WARNING:begin
                if(goback==1'b1)
                    state<=INTER;
                else
                    state<=WARNING;
            end
            RUN:begin
                if(barrier==1'b1)
                    state<=WARNING;
                else if(stop==1'b1&&excute==1'b1)
                    state<=SEARCH;
                else
                    state<=RUN;
            end
            SEARCH:begin
                if(done==1'b1)
                    state<=INTER;
                else
                    state<=SEARCH;
            end
            default:begin
                state<=START;
            end
        endcase
    end
end

/*************************************************************/
//data in state machine
/*************************************************************/

//barrier
always @(posedge clk) begin
```

```
    if(!I_rst_n)
        barrier<=1'b0;
    else if(rxdstate[7])//D?
        barrier<=1'b1;
    else
        barrier<=1'b0;
end

//goback
always @(posedge clk) begin
    if(!I_rst_n)
        goback<=1'b0;
    else if(rxdstate[0])//b
        goback<=1'b1;
    else
        goback<=1'b0;
end
//direction
always @(posedge clk) begin
    if(!I_rst_n)
        direction<=1'b0;
    else if(rxdstate[0]||rxdstate[1]||rxdstate[2]||rxdstate[3])//f l r b
        direction<=1'b1;
    else
        direction<=1'b0;
end
//stop
always @(posedge clk) begin
    if(!I_rst_n)
        stop<=1'b0;
    else if(rxdstate[4])//s
        stop<=1'b1;
    else
        stop<=1'b0;
end
//done
always @(posedge clk) begin
    if(!I_rst_n)
        done<=1'b0;
    else if(rxdstate[5])//e
        done<=1'b1;
    else
        done<=1'b0;
end
```

```
/*************************************************************/
//set uart's data
/*************************************************************/
//rdn
always @(posedge clk) begin
    if(!I_rst_n)begin
        rdn<=1'b1;
    end
    else begin
        if(r_ready)
            rdn<=1'b0;
        else
            rdn<=1'b1;
    end
end
//wrn
reg searchrecord;
reg Forwardrecord;
reg Backrecord;
reg Stoprecord;
reg Leftrecord;
reg Rightrecord;
always @(posedge clk) begin
    if(!I_rst_n)begin
        wrn<=1'b1;
        searchrecord<=1'b1;
        Forwardrecord<=1'b1;
        Backrecord<=1'b1;
        Stoprecord<=1'b1;
        Leftrecord<=1'b1;
        Rightrecord<=1'b1;
    end
    else if(state==SEARCH)begin            //SEARCH
        if(t_empty)
            if(rxdstate!=Inii&&searchrecord)begin        //only transmite when not get
i
                wrn<=1'b0;
                searchrecord<=1'b0;
                Forwardrecord<=1'b1;
                Backrecord<=1'b1;
                Stoprecord<=1'b1;
                Leftrecord<=1'b1;
                Rightrecord<=1'b1;
```

```
            end
            else
                wrn<=1'b1;
        else
            wrn<=1'b1;
    end
    else if(state==RUN||state==INTER)begin //RUN&INTER
        if(t_empty)begin
            if(runstate==Forward)begin
                if(rxdstate!=Forr&&Forwardrecord)begin         //transmite once
                    wrn<=1'b0;
                    searchrecord<=1'b1;
                    Forwardrecord<=1'b0;
                    Backrecord<=1'b1;
                    Stoprecord<=1'b1;
                    Leftrecord<=1'b1;
                    Rightrecord<=1'b1;
                end
                else
                    wrn<=1'b1;
            end
            else if(runstate==Back)begin
                if(rxdstate!=Bacc&&Backrecord)begin         //transmite once
                    wrn<=1'b0;
                    searchrecord<=1'b1;
                    Forwardrecord<=1'b1;
                    Backrecord<=1'b0;
                    Stoprecord<=1'b1;
                    Leftrecord<=1'b1;
                    Rightrecord<=1'b1;
                end
                else
                    wrn<=1'b1;
            end
            else if(runstate==Left)begin
                if(rxdstate!=Leff&&Leftrecord)begin        //transmite once
                    wrn<=1'b0;
                    searchrecord<=1'b1;
                    Forwardrecord<=1'b1;
                    Backrecord<=1'b1;
                    Stoprecord<=1'b1;
                    Leftrecord<=1'b0;
                    Rightrecord<=1'b1;
                end
```

```
                else
                    wrn<=1'b1;
            end
            else if(runstate==Right)begin
                if(rxdstate!=Rigg&&Rightrecord)begin         //transmite once
                    wrn<=1'b0;
                    searchrecord<=1'b1;
                    Forwardrecord<=1'b1;
                    Backrecord<=1'b1;
                    Stoprecord<=1'b1;
                    Leftrecord<=1'b1;
                    Rightrecord<=1'b0;
                end
                else
                    wrn<=1'b1;

            end
            else if(runstate==Stop)begin
                if(rxdstate!=Stoo&&Stoprecord)begin         //transmite once
                    wrn<=1'b0;
                    searchrecord<=1'b1;
                    Forwardrecord<=1'b1;
                    Backrecord<=1'b1;
                    Stoprecord<=1'b0;
                    Leftrecord<=1'b1;
                    Rightrecord<=1'b1;
                end
                else
                    wrn<=1'b1;
            end
            else
                wrn<=1'b1;
        end
        else
            wrn<=1'b1;
    end
    else if(state==WARNING)begin         //WARNING
        if(t_empty)begin
            if(runstate==Back)begin
                if(rxdstate!=Bacc&&Backrecord)begin       //transmite once
                    wrn<=1'b0;
                    searchrecord<=1'b1;
                    Forwardrecord<=1'b1;
                    Backrecord<=1'b0;
```

```
                    Stoprecord<=1'b1;
                    Leftrecord<=1'b1;
                    Rightrecord<=1'b1;
                end
                else
                    wrn<=1'b1;
            end
            else
                wrn<=1'b1;
        end
        else
            wrn<=1'b1;
    end
    else
        wrn<=1'b1;
end
//rxdstate
always @(posedge clk) begin
    if(!I_rst_n)begin
        rxdstate<=8'b00000000;
    end
    else begin
        case(d_out)
            8'b01000100:rxdstate<=Dann;//D
            8'b01101001:rxdstate<=Inii;//i
            8'b01100101:rxdstate<=Endd;//e
            8'b01110011:rxdstate<=Stoo;//s
            8'b01100110:rxdstate<=Forr;//f
            8'b01101100:rxdstate<=Leff;//l
            8'b01110010:rxdstate<=Rigg;//r
            8'b01100010:rxdstate<=Bacc;//b
            default:rxdstate<=rxdstate;
        endcase
    end
end
//d_in
always @(posedge clk) begin
    if(!I_rst_n)
        d_in<=8'b00000000;
    else if(state==SEARCH)
        d_in<=8'b01010111;              //W
    else begin
        case(runstate)
            Forward:
```

```
                d_in<=8'b01000110;      //F
            Left:
                d_in<=8'b01001100;      //L
            Right:
                d_in<=8'b01010010;      //R
            Stop:
                d_in<=8'b01010011;      //S
            Back:
                d_in<=8'b01000010;      //B
            default:d_in<=8'b00000000;  //none
        endcase
    end
end
/*************************************************************/
//other data set
/*************************************************************/
always @(posedge clk) begin
    if(!I_rst_n)
        playseven<=4'b0000;
    else begin
        case(rxdstate)
            Dann:playseven<=4'b0001;//D
            Inii:playseven<=4'b0010;//i
            Endd:playseven<=4'b0011;//e
            Stoo:playseven<=4'b0100;//s
            Forr:playseven<=4'b0101;//f
            Leff:playseven<=4'b0110;//l
            Rigg:playseven<=4'b0111;//r
            Bacc:playseven<=4'b1000;//b
            default:playseven<=4'b0000;
        endcase
    end
end
/*************************************************************/
//set traxial's data
/*************************************************************/
//running
always @(posedge clk) begin
    if(!I_rst_n)
        running<=1'b0;
    else if(state==RUN || state==WARNING || state==INTER)//get data
        running<=1'b1;
    else
        running<=1'b0;
```

```
end

/*************************************************************/
//set vga's data
/*************************************************************/
reg signed [15:0] sin ;
reg signed [15:0] cos ;
reg signed [15:0] temp_sin ;
reg signed [15:0] temp_cos ;
reg [9:0] angle_temp;
reg [3:0] quadrant;
reg [15:0] length;
reg [7:0] lastdirec;
reg signed [15:0] originalX;
reg signed [15:0] originalY;
reg [3:0] countsearch;
reg [7:0] distance;
reg [9:0] temp_angle;
reg [3:0] temp_quadrant;
parameter waringdis = 15 ;

//vga_state
always @(posedge clk) begin
    if(!I_rst_n)
        vga_state<=5'b00000;
    else
        vga_state<=state;
end

//clk may need to change
wire clkg;
divider #(.NUM_DIV(1353382)) divid1(.I_CLK(clk),.rst(!I_rst_n),.O_CLK(clkg));
//angle(1-361)
always @(posedge clkg) begin
    if(!I_rst_n)begin
        angle=10'b0000000000;//45
    end
    else begin
        if(angle==10'b0000000000)
            angle=10'b0101101000+1;//361
        else if(angle==10'b0101101000+2)
            angle=10'b0000000001;//1
        else if(rxdstate==Leff)
            angle=angle-1;
```

```
        else if(rxdstate==Rigg)
            angle=angle+1;
        else
            angle=angle;
    end
end
//clk may need to change
wire clkv;
divider #(.NUM_DIV(3000000)) divid2(.I_CLK(clk),.rst(!I_rst_n),.O_CLK(clkv));
//originalX/originalY/lastdirec/length
always @(posedge clkv) begin
    if(!I_rst_n)begin
        lastdirec=8'b00000000;
        length=16'b0000000000000000;
        originalX=300;
        originalY=400;
    end
    else begin
        if(rxdstate==Forr||rxdstate==Bacc)begin
            if(rxdstate!=lastdirec)begin
                length=16'b0000000000000000;
                lastdirec=rxdstate;
                originalX=site_X;
                originalY=site_Y;
            end
            else begin
                length=length+1;
            end
        end
        else begin
            lastdirec=rxdstate;
            originalX=site_X;
            originalY=site_Y;
            length=16'b0000000000000000;
        end
    end
end
//site_X/site_Y
always @(posedge clkv) begin
    if(!I_rst_n)begin
        site_X<=16'b0000000100101100;
        site_Y<=16'b0000000110010000;
    end
    else begin
```

```
        if(rxdstate==Forr||rxdstate==Bacc)begin//Forward & Back
            angle_temp=angle;
            if(rxdstate==Bacc)begin
                if(angle_temp>=180)begin
                    angle_temp=angle_temp-180;
                end
                else begin
                    angle_temp=angle_temp+180;
                end
            end
            /*preparation*/
            if(angle_temp>=1&&angle_temp<=90)begin
                quadrant=4'b0001;
                angle_temp=angle_temp-1;
            end
            else if(angle_temp>90&&angle_temp<=180)begin
                quadrant=4'b0010;
                angle_temp=angle_temp-90-1;
            end
            else if(angle_temp>180&&angle_temp<=270)begin
                quadrant=4'b0100;
                angle_temp=angle_temp-180-1;
            end
            else if(angle_temp>270&&angle_temp<=361)begin
                quadrant=4'b1000;
                angle_temp=angle_temp-270-1;
            end
            else begin
                quadrant=4'b0001;
                angle_temp=10'b0000000000;
            end
            /*taylor expansion*/
            sin=(angle_temp*3141/180)/1-
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)/10
00/1000/2/3+
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)/1000/1000/1000/1000/2/3/4/5-
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/18
0)/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7+
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/18
0)*(angle_temp*3141/180)*(angle_temp*3141/180)/1000/1000/1000/1000/1000/1000/1000/1
000/2/3/4/5/6/7/8/9-
```

```
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/18
0)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*31
41/180)/1000/1000/1000/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8/9/10/11;
            cos=1000-
                (angle_temp*3141/180)*(angle_temp*3141/180)/1000/2+
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)/1000/1000/1000/2/3/4-
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)/1000/1000/1000/1000
/1000/2/3/4/5/6+
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/18
0)*(angle_temp*3141/180)/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8-
                (angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(a
ngle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/18
0)*(angle_temp*3141/180)*(angle_temp*3141/180)*(angle_temp*3141/180)/1000/1000/1000
/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8/9/10;
            /*change*/
            case(quadrant)
                4'b0001:begin
                    site_X=originalX+length*sin/1000;
                    site_Y=originalY-length*cos/1000;
                end
                4'b0010:begin
                    site_X=originalX+length*cos/1000;
                    site_Y=originalY+length*sin/1000;
                end
                4'b0100:begin
                    site_X=originalX-length*sin/1000;
                    site_Y=originalY+length*cos/1000;
                end
                4'b1000:begin
                    site_X=originalX-length*cos/1000;
                    site_Y=originalY-length*sin/1000;
                end
                default: begin
                    site_X=originalX;
                    site_Y=originalY;
                end
            endcase
            /*boundary*/
            if(site_X>=600)
                 site_X=600;
```

```
            else if (site_X<=8)
                site_X=2;
            if(site_Y>=440)
                site_Y=440;
            else if (site_Y<=8)
                site_Y=2;
        end
    end
end
//broad
always @(posedge clk) begin
    if(!I_rst_n)begin
        broad=1486'b0;
        countsearch=4'b0000;
        temp_angle=16'b0;
        temp_quadrant=4'b0000;
    end
    else begin
        if(state==SEARCH)begin
                distance=8'b00111010;
                temp_angle=45;
                        /*temp_angle(There are som bugs that haven't been solved)*/
                        /*if(countsearch==0)begin
                            temp_angle=temp_angle-90;
                            if(temp_angle<1)
                            temp_angle=temp_angle+360;
                        end
                        else if(countsearch==1)begin
                            temp_angle=temp_angle-60;
                            if(temp_angle<1)
                                temp_angle=temp_angle+360;
                        end
                        else if(countsearch==2)begin
                            temp_angle=temp_angle-30;
                            if(temp_angle<1)
                                temp_angle=temp_angle+360;
                        end
                        else if(countsearch==3)begin
                            temp_angle=temp_angle;
                        end
                        else if(countsearch==4)begin
                            temp_angle=temp_angle+30;
                            if(temp_angle>361)
                                temp_angle=temp_angle-360;
```

```
                        end
                        else if(countsearch==5)begin
                            temp_angle=temp_angle+60;
                            if(temp_angle>361)
                                temp_angle=temp_angle-360;
                        end
                        else if(countsearch==6)begin
                            temp_angle=temp_angle+90;
                            if(temp_angle>361)
                                temp_angle=temp_angle-360;
                        end*/
                        /*preparation*/
                        if(angle>=1&&angle<=90)begin
                            temp_quadrant=4'b0001;
                        end
                        else if(angle>90&&angle<=180)begin
                            temp_quadrant=4'b0010;
                        end
                        else if(angle>180&&angle<=270)begin
                            temp_quadrant=4'b0100;
                        end
                        else if(angle>270&&angle<=361)begin
                            temp_quadrant=4'b1000;
                        end
                        else begin
                            temp_quadrant=4'b0001;
                        end

                        /*taylor expansion*/
                        temp_sin=(temp_angle*3141/180)/1-
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)/1000/1000/2/3+
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/1000/1000/1000/2/3/4/5-
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_an
gle*3141/180)/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7+
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_an
gle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/1000/1000/1000/1000/
1000/1000/1000/2/3/4/5/6/7/8/9-
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_an
gle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(te
```

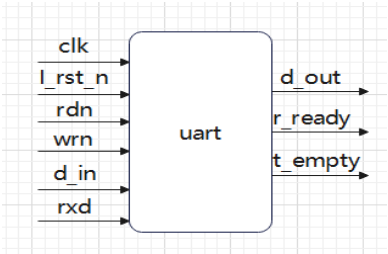
```
mp_angle*3141/180)/1000/1000/1000/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8/
9/10/11;
                        temp_cos=1000-
                            (temp_angle*3141/180)*(temp_angle*3141/180)/1000/2+
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)/1000/1000/1000/2/3/4-
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/100
0/1000/1000/1000/2/3/4/5/6+
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_an
gle*3141/180)*(temp_angle*3141/180)/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/
8-
                            (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3
141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_an
gle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/100
0/1000/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8/9/10;

                        /*calculate*/
                        case(temp_quadrant)
                            4'b0001:broad[(site_X+distance*temp_sin/1000)/13+(site_Y-
distance*temp_cos/1000)/13*45]=1'b1;
                            4'b0010:broad[(site_X+distance*temp_cos/1000)/13+(site_Y+
distance*temp_sin/1000)/13*45]=1'b1;
                            4'b0100:broad[(site_X-distance*temp_sin/1000)/13+(site_Y+
distance*temp_cos/1000)/13*45]=1'b1;
                            4'b1000:broad[(site_X-distance*temp_cos/1000)/13+(site_Y-
distance*temp_sin/1000)/13*45]=1'b1;
                            default: ;
                        endcase
        end
        else if(state==WARNING)begin
            if(countsearch==0)begin
                temp_angle=angle;
                /*preparation*/
                if(temp_angle>=1&&temp_angle<=90)begin
                    temp_quadrant=4'b0001;
                    temp_angle=temp_angle-1;
                end
                else if(temp_angle>90&&temp_angle<=180)begin
                    temp_quadrant=4'b0010;
                    temp_angle=temp_angle-90-1;
                end
                else if(temp_angle>180&&temp_angle<=270)begin
```

```
                    temp_quadrant=4'b0100;
                    temp_angle=temp_angle-180-1;
                end
                else if(temp_angle>270&&temp_angle<=361)begin
                    temp_quadrant=4'b1000;
                    temp_angle=temp_angle-270-1;
                end
                else begin
                    temp_quadrant=4'b0001;
                    temp_angle=10'b0000000000;
                end

                /*taylor expansion*/
                temp_sin=(temp_angle*3141/180)/1-
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)/1000/1000/2/3+
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/1000/1000/1000/2/3/4/5-
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*314
1/180)/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7+
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*314
1/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/1000/1000/1000/1000/1000/10
00/1000/2/3/4/5/6/7/8/9-
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*314
1/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angl
e*3141/180)/1000/1000/1000/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8/9/10/11
;
                temp_cos=1000-
                    (temp_angle*3141/180)*(temp_angle*3141/180)/1000/2+
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)/1000/1000/1000/2/3/4-
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/1000/1000/
1000/1000/2/3/4/5/6+
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*314
1/180)*(temp_angle*3141/180)/1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8-
                    (temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180
)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*314
1/180)*(temp_angle*3141/180)*(temp_angle*3141/180)*(temp_angle*3141/180)/1000/1000/
1000/1000/1000/1000/1000/1000/1000/2/3/4/5/6/7/8/9/10;
```

```
                /*calculate*/
                case(temp_quadrant)
                    4'b0001:broad[(site_X+waringdis*temp_sin/1000)/13+(site_Y-waring
dis*temp_cos/1000)/13*45]=1'b1;
                    4'b0010:broad[(site_X+waringdis*temp_cos/1000)/13+(site_Y+waring
dis*temp_sin/1000)/13*45]=1'b1;
                    4'b0100:broad[(site_X-waringdis*temp_sin/1000)/13+(site_Y+waring
dis*temp_cos/1000)/13*45]=1'b1;
                    4'b1000:broad[(site_X-waringdis*temp_cos/1000)/13+(site_Y-waring
dis*temp_sin/1000)/13*45]=1'b1;
                    default: ;
                endcase

                countsearch=countsearch+1;
            end
        end
        else begin
        countsearch=4'b0000;
        end
    end
end

/*************************************************************/
//set for test
/*************************************************************/
always @(posedge clk) begin
    if(!I_rst_n)begin
        showstate<=5'b00000;
    end
    else begin
        showstate<= state;
    end
end

endmodule
```

2. uart 模块

将 uart_receiver 和 uart_transmitter 模块进行封装，实现一个整体模块。

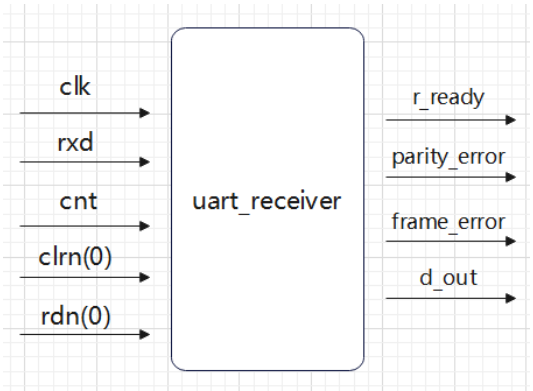


```
`timescale 1ns / 1ps
module uart(
    input clk,              //100Mhz
    input I_rst_n,          //low level effective
    input rdn,              //low level effective
    input wrn,              //low level effectives
    input rxd,
    input [7:0]d_in,
    output[7:0]d_out,
    output r_ready,
    output t_empty,
    output txd
    );

/*************************************************************/
// inter signals
/*************************************************************/
parameter count=651;
reg [15:0] num;
reg [3:0] cnt;
reg clkin;
wire parity_error;     //ignore
wire frame_error;      //ignore

/*************************************************************/
// produce clock--clkin
/*************************************************************/
always@(posedge clk or negedge I_rst_n)
begin
    if(!I_rst_n)begin
        num<=16'b0000000000000000;
        clkin<=1'b0;
    end
    else begin
        num=num+16'b0000000000000001;
        if(num>=count/2) begin
            clkin=~clkin;
            num<=16'b0000000000000000;
        end
    end
end

/*************************************************************/
// produce cnt
```

```
/*************************************************************/
always @(posedge clkin or negedge I_rst_n) begin
    if(!I_rst_n)begin
        cnt<=4'b0000;
    end
    else begin
        cnt <= cnt + 4'b0001;
    end
end

/*************************************************************/
//module part
/*************************************************************/
//reciever
uart_receiver rx(
    .clk(clkin),.clrn(I_rst_n),.cnt(cnt),.rdn(rdn),.rxd(rxd),
    .d_out(d_out),.r_ready(r_ready),.parity_error(parity_error),.frame_error(frame_e
rror)
);
//transmitter
uart_transmitter tx(
    .clk(clkin),.clrn(I_rst_n),.wrn(wrn),.d_in(d_in),.cnt(cnt),
    .t_empty(t_empty),.txd(txd)
);
endmodule
```

3. uart_receiver 模块

通过串口协议进行双蓝牙通信，作为 FPGA 的信息接收端。




```
`timescale 1ns / 1ps
module uart_receiver(
    input clk,                      //9600hz*16
    input clrn,                     //Low level effective
    input [3:0] cnt,                //count for 16
    input rdn,                      //Low level effective
    input rxd,                      //receibe data line
```

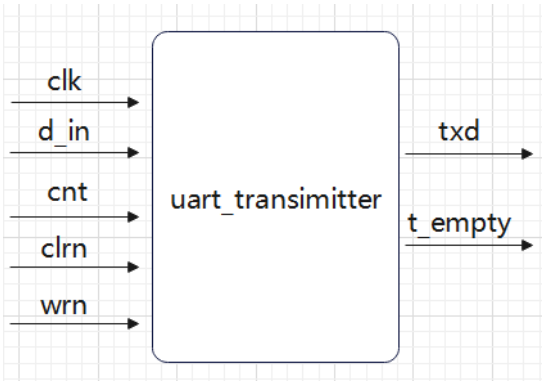
```
    output reg r_ready,             //judge whether we can read(1:can/0:can't)
    output [7:0] d_out,             //get data from rxd
    output reg parity_error=1'b1,   //1:parity_site error
    output reg frame_error          //1:end_site error
);


parameter bits=4'b1010;
//internl singnal
reg [3:0] sampling_place;
reg [3:0] no_bits;
reg [9:0] r_buffer;            //stop,data[7:0],start
reg clkr;
reg rxd_old,rxd_new;
reg sampling;
reg [7:0] frame;

/*************************************************************/
//latch 2 sampling bits
/*************************************************************/
always @ (posedge clk or negedge clrn)
begin
    if(clrn == 0) begin
        rxd_old <= 1'b1;
        rxd_new <= 1'b1;
    end
    else begin
        rxd_old <= rxd_new;
        rxd_new <= rxd;
    end
end

/*************************************************************/
//detect start bit
/*************************************************************/
always @(posedge clk or negedge clrn)
begin
    if(clrn==0)begin
        sampling <= 1'b0;
    end
    else begin
        if(rxd_old && !rxd_new) begin
            if(!sampling)
                sampling_place<= cnt+4'b1000;//half
```

```
            sampling <= 1'b1;
        end
        else begin
            if(no_bits == bits)
                sampling <= 1'b0;
        end
    end
end

/*************************************************************/
//sampling clock:clkr
/*************************************************************/
always @ (posedge clk or negedge clrn)
begin
    if(clrn == 0) begin
        clkr<=1'b0;
    end else
    begin
        if(sampling) begin
            if(cnt == sampling_place)
                clkr<=1'b1;
            if(cnt == sampling_place+4'b0001)
                clkr<=1'b0;
        end else
            clkr<= 1'b0;
    end
end

/*************************************************************/
//number of bits received
/*************************************************************/
always@(posedge clkr or negedge sampling)
begin
    if(!sampling) begin
        no_bits<=4'b0000;
    end
    else begin
        no_bits<=no_bits + 4'b0001;
        r_buffer[no_bits]<= rxd;
    end
end

/*************************************************************/
//data processing
```

```
/*************************************************************/
always @(posedge clk or negedge clrn or negedge rdn)
begin
    if(clrn==0) begin
        r_ready <= 1'b0;
        frame_error <=1'b0;
    end
    else begin
        if(!rdn) begin
            r_ready <= 1'b0;
            frame_error <= 1'b0;
        end
        else begin
            if(no_bits == bits) begin
            frame <= r_buffer[8:1];
            r_ready <= 1'b1;
            if(!r_buffer[9]) begin
                frame_error<=1'b1;
            end
            end
        end
    end
end

assign d_out =!rdn ? frame :8'bz;
endmodule
```

4. uart_transmitter 模块

通过串口协议进行双蓝牙通信，作为 FPGA 的信息发送端。



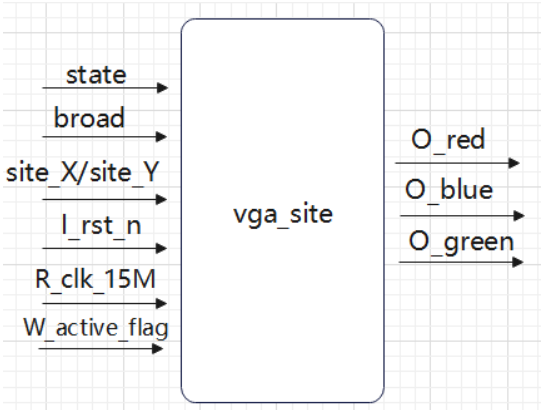

```
`timescale 1ns / 1ps
module uart_transmitter (
    input clk,                  //9600hz*16
    input clrn,                 //Low level effective
    input wrn,                  //Low level effective
    input [7:0] d_in,           //put data in the txd
```

```
    input [3:0] cnt,            //count 16
    output reg txd,             //txd line
    output reg t_empty          //judge whether we can put data in(1:can/0:can't)
);

parameter bits=4'b1010;
//internal singnals
reg [3:0] no_bits;
reg [7:0] t_buffer;         //data_buffer
reg sending;                //transport_buffer
reg [7:0] d_buffer;
reg load_t_buffer;

/*************************************************************/
//load d_in ,sending enalbe , t_empty ,sending_place
/*************************************************************/
always @(posedge clk or negedge clrn or negedge wrn) begin
    if(clrn == 0)begin
        sending <= 1'b0;
        t_empty <= 1'b1;
        load_t_buffer <= 1'b0;
    end
    else begin
        if(!wrn)begin
            d_buffer <= d_in;
            t_empty <= 1'b0;
            load_t_buffer <= 1'b1;
        end
        else begin
            if(!sending)begin
                if(load_t_buffer)begin
                    sending <= 1'b1;
                    t_buffer <= d_buffer;
                    t_empty <= 1'b1;
                    load_t_buffer <= 1'b0;
                end
            end
            else begin
                if(no_bits == bits)
                sending <= 1'b0;
            end
        end
    end
end
```

```
assign clkw = cnt[3];

/*************************************************************/
//number of bits sent
/*************************************************************/
always @(posedge clkw or negedge sending) begin
    if(!sending)begin
        no_bits<=4'b0000;
        txd <= 1'b1;
    end
    else begin
        case (no_bits)
             0:txd <= 1'b0;//start
             1:txd <= t_buffer[0];
             2:txd <= t_buffer[1];
             3:txd <= t_buffer[2];
             4:txd <= t_buffer[3];
             5:txd <= t_buffer[4];
             6:txd <= t_buffer[5];
             7:txd <= t_buffer[6];
             8:txd <= t_buffer[7];
             //9:txd <= ^t_buffer;//parity site-->don't need parity
             default: txd <= 1'b0;//end
        endcase
        no_bits <= no_bits+4'b0001;
    end
end
endmodule
```

5. vga_site 模块

根据输入的小车 xy 坐标和背景障碍物信息，以及状态信息，返回 vga 时序下界面的显示情况。




```
`timescale 1ns / 1ps
module vga_site (
    input         [4:0]     state,           //search-warning-run-inter-start
```

```
    input          [1485:0]  broad,           //60*44
    input    signed[15:0]    site_X,
    input    signed[15:0]    site_Y,
    input                    I_rst_n,         //low level effective
    input                    R_clk_25M,
    input                    W_active_flag,
    input          [9:0]     R_h_cnt,
    input          [9:0]     R_v_cnt,
    output   reg   [3:0]    O_red   , // VGA
    output   reg   [3:0]    O_green , // VGA
    output   reg   [3:0]    O_blue    // VGA
);
parameter linerow = 20  ;
parameter linevolumn = 20 ;

/*************************************************************************/
//
//
//           *******
//           *     *
//           *     *
//           *******
//           *     *
//           *     *
//           *******
/*************************************************************************/
parameter car_red = 4'b0000 ;
parameter car_green = 4'b0111 ;
parameter car_blue =  4'b0000 ;
parameter radius =15 ;

parameter h_startsite = 144 ;        //96+48
parameter h_endsite = 784;           //96+48+640
parameter c_startsite = 35;          //2+33
parameter c_endsite = 515;           //2+33+480
reg signed [15:0] getx;
reg signed [15:0] gety;

always @(posedge R_clk_25M or negedge I_rst_n)
begin
    if(!I_rst_n)begin
            O_red   <=  4'b0000    ;
            O_green <=  4'b0000    ;
            O_blue  <=  4'b0000    ;
```

```
    end
    else if(W_active_flag)begin
        /*homepage*/
        if(state[0])begin
             getx=R_h_cnt-h_startsite;
             gety=R_v_cnt-c_startsite;
             if(((getx-200)*(getx-200)+(gety-180)*(gety-180))<=25*25)begin
                 O_red   <=  4'b0000   ;
                 O_green <=  4'b0000   ;
                 O_blue  <=  4'b0000   ;
             end
             else if(((getx-400)*(getx-400)+(gety-180)*(gety-180))<=25*25)begin
                 O_red   <=  4'b0000   ;
                 O_green <=  4'b0000   ;
                 O_blue  <=  4'b0000   ;
             end
             else if(((getx-300)*(getx-300)+(gety-300)*(gety-300))<=25*25)begin
                 O_red   <=  4'b0000   ;
                 O_green <=  4'b0000   ;
                 O_blue  <=  4'b0000   ;
             end
             else if(((getx-300)*(getx-300)+(gety-400)*(gety-400))<=25*25)begin
                 O_red   <=  4'b0000   ;
                 O_green <=  4'b0000   ;
                 O_blue  <=  4'b0000   ;
             end
             else if(gety<=40)begin
                 O_red   <=  4'b0101   ;
                 O_green <=  4'b0111   ;
                 O_blue  <=  4'b0000   ;
             end
             else if(gety<=80)begin
                 O_red   <=  4'b0101    ;
                 O_green <=  4'b0010    ;
                 O_blue  <=  4'b0101    ;
             end
             else if(gety<=120)begin
                 O_red   <=  4'b0001    ;
                 O_green <=  4'b0100    ;
                 O_blue  <=  4'b0111    ;
             end
             else if(gety<=160)begin
                 O_red   <=  4'b0001    ;
                 O_green <=  4'b1000    ;
```

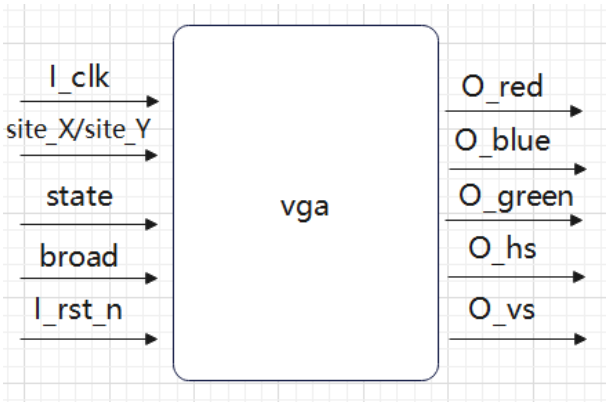
```
                O_blue  <=  4'b0110    ;
            end
            else if(gety<=200)begin
                O_red   <=  4'b0001    ;
                O_green <=  4'b0100    ;
                O_blue  <=  4'b0100    ;
            end
            else if(gety<=240)begin
                O_red   <=  4'b0011    ;
                O_green <=  4'b0110    ;
                O_blue  <=  4'b0010    ;
            end
            else if(gety<=280)begin
                O_red   <=  4'b0101    ;
                O_green <=  4'b1100    ;
                O_blue  <=  4'b0110    ;
            end
            else if(gety<=320)begin
                O_red   <=  4'b0101    ;
                O_green <=  4'b0000    ;
                O_blue  <=  4'b0000    ;
            end
            else if(gety<=360)begin
                O_red   <=  4'b1111    ;
                O_green <=  4'b0100    ;
                O_blue  <=  4'b0011    ;
            end
            else if(gety<=400)begin
                O_red   <=  4'b0000    ;
                O_green <=  4'b0110    ;
                O_blue  <=  4'b1000    ;
            end
            else begin
                O_red   <=  4'b0000    ;
                O_green <=  4'b1000    ;
                O_blue  <=  4'b0010    ;
            end
        end
        /*mainpage*/
        else if(state[1]||state[2]||state[4])begin
            getx=R_h_cnt-h_startsite;
            gety=R_v_cnt-c_startsite;
            //frame
            if((getx <= linerow || getx >= (h_endsite - linerow - h_startsite )) ||
```

```
                (gety <= linevolumn || gety >= (c_endsite - linevolumn -
c_startsite)))begin
                O_red   <=  4'b0101    ;
                O_green <=  4'b0000    ;
                O_blue  <=  4'b0000    ;
            end
            //car
            else
if(((getx-site_X)*(getx-site_X)+(gety-site_Y)*(gety-site_Y))<=radius*radius)begin
                O_red   <=  car_red     ;
                O_green <=  car_green   ;
                O_blue  <=  car_blue    ;
            end
            //barrier
            else if(broad[getx/13+gety/13*45])begin
                O_red   <=  4'b1010    ;
                O_green <=  4'b0000    ;
                O_blue  <=  4'b1111    ;
            end
            //none
            else begin
                O_red   <=  4'b0000   ;
                O_green <=  4'b0000   ;
                O_blue  <=  4'b0000   ;
            end
        end
        /*warning page*/
        else if(state[3])begin
            getx=R_h_cnt-h_startsite;
            gety=R_v_cnt-c_startsite;
            //frame
            if((getx <= linerow|| getx >= (h_endsite - linerow-h_startsite)) ||
                (gety <= linevolumn || gety >= (c_endsite -
linevolumn-c_startsite)))begin
                O_red   <=  4'b0101    ;
                O_green <=  4'b1000    ;
                O_blue  <=  4'b0000    ;
             end
            //Waiting for the supplement...

            //car
            else
if(((getx-site_X)*(getx-site_X)+(gety-site_Y)*(gety-site_Y))<=radius*radius)begin
                O_red   <=  car_red     ;
```

```
                O_green <=  car_green   ;
                O_blue  <=  car_blue    ;
            end
            //barrier
            else if(broad[getx/13+gety/13*45])begin
                O_red   <=  4'b1010    ;
                O_green <=  4'b0000    ;
                O_blue  <=  4'b1111    ;
            end
            //none
            else begin
                O_red   <=  4'b0000   ;
                O_green <=  4'b0000   ;
                O_blue  <=  4'b0000   ;
            end

        end
    end
    else begin
        O_red   <=  4'b0000    ;
        O_green <=  4'b0000    ;
        O_blue  <=  4'b0000    ;
    end
end
endmodule
```

6. vga 模块

连接 vga_site 模块，形成独立的 vga 功能模块。根据小球坐标、背景障碍物信息以及状态信息，在 vga 上显示实时定位信息和环境检测情况。




```
`timescale 1ns / 1ps
module vga(
    input    signed [15:0]   site_X  ,
    input    signed [15:0]   site_Y  ,
    input           [4 :0]   state   ,   // search-warning-run-inter-start   high level
effective
```

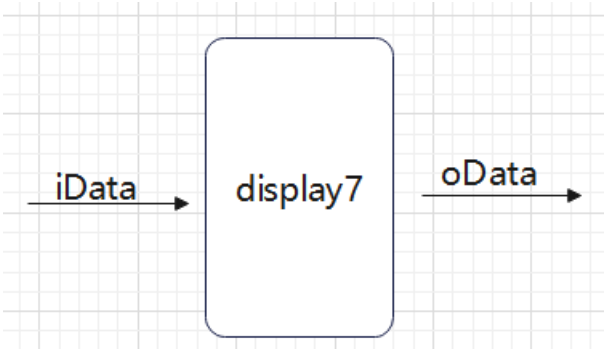
```
    input           [1485:0] broad   ,   // 60*44
    input                    I_clk   ,   // 100Mhz-clock
    input                    I_rst_n ,   // low level effective
    output   wire   [3:0]    O_red   ,   // VGA's red   channel
    output   wire   [3:0]    O_green ,   // VGA's green channel
    output   wire   [3:0]    O_blue  ,   // VGA's blue  channel
    output                   O_hs    ,
    output                   O_vs
);

// 640*480
parameter      C_H_SYNC_PULSE      =   96  ;
parameter      C_H_BACK_PORCH      =   48  ;
parameter      C_H_ACTIVE_TIME     =   640 ;
parameter      C_H_FRONT_PORCH     =   16  ;
parameter      C_H_LINE_PERIOD     =   800 ;

// 640*480
parameter      C_V_SYNC_PULSE      =   2   ;
parameter      C_V_BACK_PORCH      =   33  ;
parameter      C_V_ACTIVE_TIME     =   480 ;
parameter      C_V_FRONT_PORCH     =   10  ;
parameter      C_V_FRAME_PERIOD    =   525 ;

reg [9:0]       R_h_cnt         ; //
reg [9:0]       R_v_cnt         ; //
wire            R_clk_25M       ; //
wire            W_active_flag   ; //

/*****************************************************************/
//produce a 25Mhz-clock
/*****************************************************************/
divider #(.NUM_DIV(4)) div(.I_CLK(I_clk),.rst(!I_rst_n),.O_CLK(R_clk_25M));

/*****************************************************************/
//produce O_hs
/*****************************************************************/
always @(posedge R_clk_25M or negedge I_rst_n)
begin
    if(!I_rst_n)
        R_h_cnt <=  9'd0   ;
    else if(R_h_cnt == C_H_LINE_PERIOD - 1'b1)
        R_h_cnt <=  9'd0   ;
    else
```

```
        R_h_cnt <=  R_h_cnt + 1'b1  ;
end
assign O_hs =   (R_h_cnt < C_H_SYNC_PULSE||!I_rst_n) ? 1'b0 : 1'b1    ;

/***********************************************************************/
//produce O_vs
/***********************************************************************/
always @(posedge R_clk_25M or negedge I_rst_n)
begin
    if(!I_rst_n)
        R_v_cnt <=  9'd0   ;
    else if(R_v_cnt == C_V_FRAME_PERIOD - 1'b1)
        R_v_cnt <=  9'd0   ;
    else if(R_h_cnt == C_H_LINE_PERIOD - 1'b1)//琛屾椂搴忔弧浜?--鍣嬪満鐮跺簭鍔?1
        R_v_cnt <=  R_v_cnt + 1'b1  ;
    else
        R_v_cnt <=  R_v_cnt ;
end
assign O_vs =   (R_v_cnt < C_V_SYNC_PULSE||!I_rst_n) ? 1'b0 : 1'b1    ;

/***********************************************************************/
//produce W_active_flag
/***********************************************************************/
assign W_active_flag =  (R_h_cnt >= (C_H_SYNC_PULSE +
C_H_BACK_PORCH                  ))  &&
                       (R_h_cnt <= (C_H_SYNC_PULSE + C_H_BACK_PORCH +
C_H_ACTIVE_TIME))  &&
                       (R_v_cnt >= (C_V_SYNC_PULSE +
C_V_BACK_PORCH                  ))  &&
                       (R_v_cnt <= (C_V_SYNC_PULSE + C_V_BACK_PORCH +
C_V_ACTIVE_TIME))  ;

/***********************************************************************/
//module for show
/***********************************************************************/
vga_site site(
    .state(state),
    .broad(broad),
    .site_X(site_X),
    .site_Y(site_Y),
    .I_rst_n(I_rst_n),
    .R_clk_25M(R_clk_25M),
    .W_active_flag(W_active_flag),
    .R_h_cnt(R_h_cnt),
```

```
    .R_v_cnt(R_v_cnt),
    .O_red(O_red)   ,    // VGA
    .O_green(O_green) ,  // VGA
    .O_blue(O_blue)      // VGA
);
endmodule
```

7. display7 模块

根据 uart 串口的信息获取情况，在七位数码管上显示。



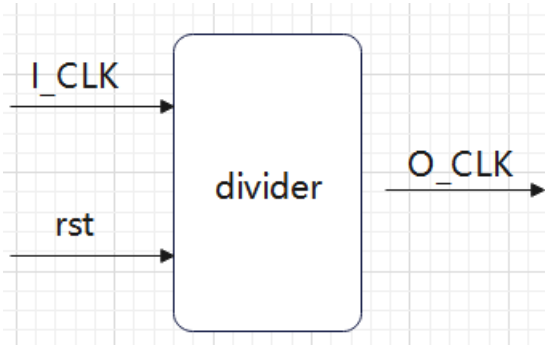
七段数码管显示情况信息：

1	2	3	4	5	6	7	8
D	i	e	s	f	l	r	b
遇障信号	探测开始	探测结束	停止	前进	左转	右转	后退

```
`timescale 1ns / 1ps
module display7(iData,oData);
       input [3:0] iData;
       output reg [6:0] oData;
always@(*)
begin
     case(iData)
         4'b0000:oData=7'b1000000;
         4'b0001:oData=7'b1111001;
         4'b0010:oData=7'b0100100;
         4'b0011:oData=7'b0110000;
         4'b0100:oData=7'b0011001;
         4'b0101:oData=7'b0010010;
         4'b0110:oData=7'b0000010;
         4'b0111:oData=7'b1111000;
         4'b1000:oData=7'b0000000;
         4'b1001:oData=7'b0010000;
         default: oData=7'b1000000;
     endcase
end
endmodule
```

8. divider 模块

对系统时钟进行分频，以满足不同的时序需求。

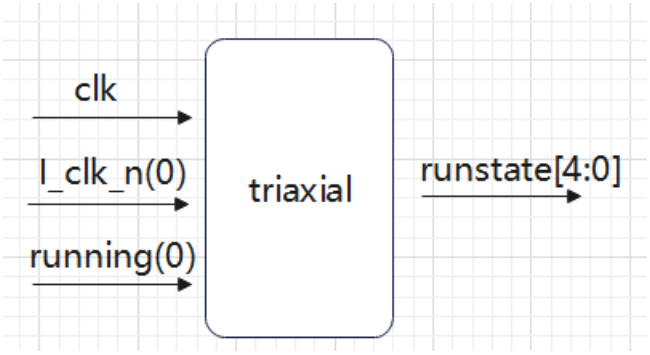



```
`timescale 1ns / 1ps
module divider(I_CLK,rst,O_CLK);
    input I_CLK;              //输入时钟限号 上升沿有效
    input rst;                //同步复位信号 高电平有效
    output reg O_CLK;         //输出时钟
    integer count;
    integer first=1'b0;
parameter NUM_DIV=20;
//初始化输出
initial
begin
   if(first==0)
     begin
     O_CLK=1'b0;
     first=first+1;
     count=1'b0;
     end
end
//进行改变
always @ (posedge I_CLK)
begin
    count=count+1;//使用阻塞赋值
    if(rst)//同步复位信号--高电平有效
        begin
        O_CLK=1'b0;
        count=0;
        end
    else if(count==NUM_DIV/2)
        begin
        O_CLK=~O_CLK;
        count=0;
        end
    else
```

```
        O_CLK=O_CLK;
end
endmodule
```

9. triaxial 模块

本模块负责对输入信号进行接受，处理后返回控制器。




```
`timescale 1ns / 1ps
module triaxial (
    input clk,
    input I_rst_n,              //low level effective
    input [4:0] set,            //will change that...
    input running,              //high level effective
    output reg [4:0] runstate  //F10000 L01000 S00100 R00010 B00001
);

always @(posedge clk) begin
    if(!I_rst_n)begin
        runstate<=5'b00000;
    end
    else if(!running)begin
        runstate<=5'b00000;
    end
    else begin
        if(set[0])begin
            runstate<=5'b00001;
        end
        else if(set[1])begin
            runstate<=5'b00010;
        end
        else if(set[2])begin
            runstate<=5'b00100;
        end
        else if(set[3])begin
            runstate<=5'b01000;
        end
        else if(set[4])begin
```

```
            runstate<=5'b10000;
        end
        else begin
            runstate<=5'b00000;
        end
    end
end
//wait for change...
endmodule
```

本模块最开始基于 SPI 通信的 L3G4200D 三轴加速器对行驶信号进行采集，但后面由于 L3G4200D 信息不全，调试困难，难以获得较好的控制信号，受限于项目提交时间限制，故弃用，改为使用 FPGA 按钮进行控制。以下给出 L3G4200D 模块代码。

```
`timescale 1ns / 1ps
module SPI_transmitter(
    input       clk,
    input       rst,                //high

    // SPI port
    output reg CSN,                 //set 0 for SPI
    output reg SCLK,
    output reg MOSI,                //涓绘満权撳嚭
    input       MISO,               //涓绘満权撳緳

    // Control port
    input             ready,        //1 --> begin to transmit
    input      [7:0] inst,          //write 0x0B ; read 0x0A
    input             rdh_wrl,      //1 read; 0 write
    input      [7:0] reg_addr,      //reg addr
    input      [7:0] dout,          //data input
    output reg [7:0] din,           //data back
    output reg       din_valid      //judge whether the back data is ready
);
/*****************************************************************************/
// internal signals
/*****************************************************************************/
parameter divider = 8'd05 ;
reg         SCLK_en;
reg         SCLK_d;
reg  [7:0]  SCLK_count;
wire        SCLK_posedge;
wire        SCLK_negedge;
/*****************************************************************************/
// SCK generator, 10MHz output
/*****************************************************************************/
```

```
always @(posedge clk or posedge rst) begin
     if(rst || ~SCLK_en) begin
          SCLK <= 1'b0;
         SCLK_count <= 8'd0;
     end
     else if(SCLK_en && (SCLK_count<divider)) begin
         SCLK_count <= SCLK_count + 8'd1;
     end
    else begin
         SCLK <= ~SCLK;
         SCLK_count <= 8'd0;
    end
end
/*****************************************************************************/
// dectect the high and low of the clk
/*****************************************************************************/
always @(posedge clk) begin
    SCLK_d <= SCLK;
end
assign SCLK_posedge = ({SCLK_d, SCLK}==2'b01) ? 1'b1 : 1'b0;
assign SCLK_negedge = ({SCLK_d, SCLK}==2'b10) ? 1'b1 : 1'b0;

/*****************************************************************************/
// Ready rising edge detection
/*****************************************************************************/
reg  ready_d;
wire ready_posedge;
always @(posedge clk) begin
    ready_d <= ready;
end
assign ready_posedge = ({ready_d, ready} == 2'b01) ? 1'b1 : 1'b0;
/*****************************************************************************/
// State machine
/*****************************************************************************/
reg  [3:0]  state;
reg  [3:0]  next_state;

parameter IDLE       = 4'd0;
parameter START      = 4'd1;
parameter INST_OUT   = 4'd2;
parameter ADDR_OUT   = 4'd3;
parameter WRITE_DATA = 4'd4;
parameter READ_DATA  = 4'd5;
parameter ENDING     = 4'd6;
```

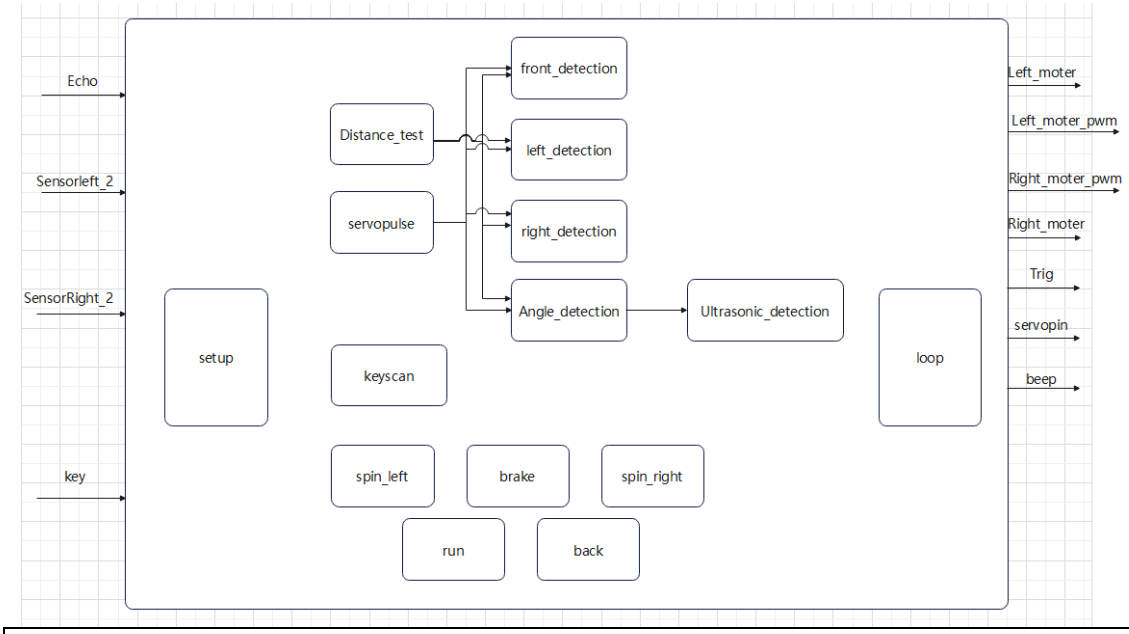
```
reg  [6:0]  MISO_buf;
reg  [7:0]  MOSI_buf;
reg  [3:0]  MOSI_count;
always @(posedge clk or posedge rst) begin
     if(rst) begin
          state <= IDLE;
     end
     else begin
          state <= next_state;
     end
end
always @(posedge clk) begin
     case(state)
     IDLE:
     begin    // IDLE state
        next_state <= START;
         MOSI <= 1'b0;
        CSN <= 1'b1;
        SCLK_en <= 1'b0;
        MOSI_buf <= inst;
        MOSI_count <= 4'd0;
        din <= 8'h00;
        din_valid <= 1'b0;
     end
     START:
     begin    // enable SCK and CS
        // start the process when ready rise, load instruction
        if(ready_posedge) begin
            next_state <= INST_OUT;
            CSN  <= 1'b0;
            SCLK_en <= 1'b1;
            MOSI_buf <= {inst[6:0], 1'b0};
            MOSI <= inst[7];
        end
     end
    INST_OUT:
     begin    // send out instruction
          if(SCLK_negedge && (MOSI_count < 4'd7)) begin
               {MOSI, MOSI_buf} <= {MOSI_buf, 1'b0};
            MOSI_count <= MOSI_count + 4'd1;
          end
          else if(SCLK_negedge) begin
               {MOSI, MOSI_buf} <= {reg_addr, 1'b0};
            MOSI_count <= 4'd0;
```

```
            next_state <= ADDR_OUT;
        end
    end
    ADDR_OUT:
    begin    // send out register address
        if(SCLK_negedge && (MOSI_count < 4'd7)) begin
            {MOSI, MOSI_buf} <= {MOSI_buf, 1'b0};
          MOSI_count <= MOSI_count + 4'd1;
        end
        else if(SCLK_negedge) begin
            {MOSI, MOSI_buf} <= {dout, 1'b0};
          MOSI_count <= 4'd0;
          next_state <= (rdh_wrl) ? READ_DATA : WRITE_DATA;
        end
    end
   WRITE_DATA:
    begin    // send testing data out
        if(SCLK_negedge && (MOSI_count < 4'd7)) begin
            {MOSI, MOSI_buf} <= {MOSI_buf, 1'b0};
          MOSI_count <= MOSI_count + 4'd1;
        end
        else if(SCLK_negedge) begin
            {MOSI, MOSI_buf} <= 9'h0;
          MOSI_count <= 4'd0;
          next_state <= ENDING;
        end
    end
    READ_DATA:
    begin    // get a byte
        if(SCLK_posedge && (MOSI_count < 4'd7)) begin
            MISO_buf <= {MISO_buf[5:0], MISO};
          MOSI_count <= MOSI_count + 4'd1;
        end
        else if(SCLK_posedge) begin
          MOSI_count <= 4'd0;
          next_state <= ENDING;
          din <= {MISO_buf, MISO};
          din_valid <= 1'b1;
        end
       else begin
          din_valid <= 1'b0;
       end
    end
    ENDING:
```

```
    begin    //disable SCK and CS
        if(SCLK_negedge) begin
            CSN <= 1'b1;
            next_state <= IDLE;
            SCLK_en <= 1'b0;
        end
    end
    endcase
end
endmodule
```

10.myrun 模块(arduino)

本模块在小车上实现，包含了各种小车操作信息：基本行驶、红外霍尔传感器探测、超声波探测等。以下为模块内部函数划分：




```
/*slep*/
/*****************************参数初始化*************************************/
char getstr;                    // 存储蓝牙串口获取的 8biti 信息
const int Left_motor=8;         // 左电机(IN3) 输出 0  前进   输出 1 后退
const int Left_motor_pwm=9;     // 左电机 PWM 调速
const int Right_motor_pwm=10;   // 右电机 PWM 调速
const int Right_motor=11;       // 右电机后退(IN1)  输出 0  前进   输出 1 后退
const int key=A2;               // 定义按键 数字 A2 接口
const int beep=A3;              // 定义蜂鸣器 数字 A3 接口
const int Echo = A1;            // Echo 回声脚(P2.0)
const int Trig =A0;             // Trig 触发脚(P2.1)
const int servopin=2;           // 设置舵机驱动脚到数字口 2
const int SensorRight_2 = 5;     // 左边红外避障传感器()
const int SensorLeft_2 = 6;      // 右边红外避障传感器()
int Front_Distance = 0;         // 前探测
```

```
int Left_Distance = 0;          // 左探测
int Right_Distance = 0;         // 右探测
int Angle_Distance=0;           // 每一角度探测的值

void setup() {
   Serial.begin(9600);            // 蓝牙波透率

   /*电机端口*/
   /*LOW 前进**HIGH 后退*/
   pinMode(Left_motor,OUTPUT);     // PIN 8   8 脚无 PWM 功能
   pinMode(Left_motor_pwm,OUTPUT); // PIN 9   (PWM)
   pinMode(Right_motor_pwm,OUTPUT);// PIN 10  (PWM)
   pinMode(Right_motor,OUTPUT);    // PIN 11  (PWM)
   /*超声波端口*/
   pinMode(Echo, INPUT);           // 定义超声波输入脚
   pinMode(Trig, OUTPUT);          // 定义超声波输出脚
   pinMode(servopin,OUTPUT);       // 设定舵机接口为输出接口
   /*红外避障端口*/
   pinMode(SensorLeft_2, INPUT);   // 定义中间避障传感器为输入
   pinMode(SensorRight_2, INPUT);  // 定义中间避障传感器为输入
   /*其余端口初始化*/
   pinMode(key,INPUT);             // 定义按键接口为输入接口
   pinMode(beep,OUTPUT);           // 蜂鸣器

}

/*****************************电机驱动函数*********************************/
/*
*@name:run
*@function:小车前进
*/
void run()
{
  digitalWrite(Right_motor,LOW);              // 右电机前进
  digitalWrite(Right_motor_pwm,HIGH);         // 右电机前进
  analogWrite(Right_motor_pwm,80);           // PWM 比例 0~255 调速，左右轮差异略增减

  digitalWrite(Left_motor,LOW);               // 左电机前进
  digitalWrite(Left_motor_pwm,HIGH);          // 左电机 PWM
  analogWrite(Left_motor_pwm,80);            // PWM 比例 0~255 调速，左右轮差异略增减

  //delay(time * 100);                        // 执行时间，可以调整
}
/*
```

```
*@name:brake
*@function:小车停止
*/
void brake()
{
  digitalWrite(Right_motor_pwm,LOW);           // 右电机 PWM 调速输出 0
  analogWrite(Right_motor_pwm,0);              // PWM 比例 0~255 调速，左右轮差异略增减

  digitalWrite(Left_motor_pwm,LOW);            // 左电机 PWM 调速输出 0
  analogWrite(Left_motor_pwm,0);               // PWM 比例 0~255 调速，左右轮差异略增减

  //delay(time * 100);                         // 执行时间，可以调整
}
/*
*@name:spin_left
*@function:小车左转(左轮后退，右轮前进)
*/
void spin_left()
{
  digitalWrite(Right_motor,LOW);               // 右电机前进
  digitalWrite(Right_motor_pwm,HIGH);          // 右电机前进
  analogWrite(Right_motor_pwm,80);            // PWM 比例 0~255 调速，左右轮差异略增减

  digitalWrite(Left_motor,HIGH);               // 左电机后退
  digitalWrite(Left_motor_pwm,HIGH);           // 左电机 PWM
  analogWrite(Left_motor_pwm,80);             // PWM 比例 0~255 调速，左右轮差异略增减

 // delay(time * 100);                         // 执行时间，可以调整
}
/*
*@name:spin_right
*@function:小车右转(右轮后退，左轮前进)
*/
void spin_right()
{
  digitalWrite(Right_motor,HIGH);              // 右电机后退
  digitalWrite(Right_motor_pwm,HIGH);          // 右电机 PWM 输出 1
  analogWrite(Right_motor_pwm,80);            // PWM 比例 0~255 调速，左右轮差异略增减


  digitalWrite(Left_motor,LOW);                // 左电机前进
  digitalWrite(Left_motor_pwm,HIGH);           // 左电机 PWM
  analogWrite(Left_motor_pwm,80);             // PWM 比例 0~255 调速，左右轮差异略增减
```

```
  //delay(time * 100);                          // 执行时间，可以调整
}
/*
*@name:back
*@function:小车后退
*/
void back()
{
  digitalWrite(Right_motor,HIGH);              // 右电机后退
  digitalWrite(Right_motor_pwm,HIGH);          // 右电机前进
  analogWrite(Right_motor_pwm,80);             // PWM 比例 0~255 调速，左右轮差异略增减


  digitalWrite(Left_motor,HIGH);               // 左电机后退
  digitalWrite(Left_motor_pwm,HIGH);           // 左电机 PWM
  analogWrite(Left_motor_pwm,80);              // PWM 比例 0~255 调速，左右轮差异略增减

  //delay(time * 100);                          // 执行时间，可以调整
}

/*****************************舵机测距函数*********************************/
/*
*@name:Distance_test
*@function:量出前方距离
*/
float Distance_test()
{
  digitalWrite(Trig, LOW);                     // 给触发脚低电平 2μs
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);                    // 给触发脚高电平 10μs，这里至少是 10μs
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);                     // 持续给触发脚低电
  float Fdistance = pulseIn(Echo, HIGH);       // 读取高电平时间(单位：微秒)
  Fdistance= Fdistance/58;                     // 为什么除以 58 等于厘米，  Y 米=（X 秒*344）/2
                                               //X 秒=（ 2*Y 米）/344 ==》X 秒=0.0058*Y 米 ==》
厘米=微秒/58
  return Fdistance;
}
/*
*@name:servopulse
*@function:定义一个脉冲函数，用来模拟方式产生 PWM 值舵机的范围是 0.5MS 到 2.5MS 1.5MS 占空比是
居中周期是 20MS
*@attention:不同的 PWM 波占空比对应 servo 转动的不同角度
*/
```

```
void servopulse(int servopin,int myangle)
{
  int pulsewidth=(myangle*11)+500;              //将角度转化为500-2480 的脉宽值 这里的myangle
就是0-180度  所以180*11+500=2480  11是为了换成90度的时候基本就是1.5MS
  digitalWrite(servopin,HIGH);                                  // 将 舵 机 接 口 电 平 置 高
90*11+50=1490uS  就是1.5ms
  delayMicroseconds(pulsewidth);                //延时脉宽值的微秒数  这里调用的是微秒延时函
数
  digitalWrite(servopin,LOW);                   //将舵机接口电平置低
  delay(20-(pulsewidth*0.001));                 //延时周期内剩余时间  这里调用的是ms延时函数
}
/*
*@name:front_detection
*@function:探测前方距离
*/
void front_detection()
{
  //此处循环次数减少，为了增加小车遇到障碍物的反应速度
  for(int i=0;i<=5;i++){                        //产生PWM个数，等效延时以保证能转到响应角度
    servopulse(servopin,90);                    //模拟产生PWM
  }
  Front_Distance = Distance_test();
}
/*
*@name:left_detection
*@function:探测左边距离
*/
void left_detection()
{
  for(int i=0;i<=15;i++){                       //产生PWM个数，等效延时以保证能转到响应角度
    servopulse(servopin,175);                   //模拟产生PWM
  }
  Left_Distance = Distance_test();
}
/*
*@name:right_detection
*@function:探测右边距离
*/
void right_detection()
{
  for(int i=0;i<=15;i++){                       //产生PWM个数，等效延时以保证能转到响应角度
    servopulse(servopin,5);                     //模拟产生PWM
  }
  Right_Distance = Distance_test();
```

```
}
/*
*@name:Angle_detection
*@function:探测某一角度的值
*/
void Angle_detection(int angle)
{
  int times=10;
  //转到对应的位置
  for(int i=0;i<=times;i++){                    //产生 PWM 个数，等效延时以保证能转到响应角度
    servopulse(servopin,angle);                 //模拟产生 PWM
  }
  //开始探测距离
  Angle_Distance = Distance_test();             //Distan_test 返回 float，强制转化为 int
  //发送距离信息
  if(Angle_Distance>=255)                       //将距离信息控制在 255cm 以；当为 1111 1111 时表
示距离无限远
      Angle_Distance=0xFF;
  Serial.write(Angle_Distance);
}
/*
*@name:Ultrasonic_Detection
*@function:进行 5-30-60-90-120-150-175 度的超声波探测
*/
void Ultrasonic_Detection()
{
  Angle_detection(5);
  for(int i=30;i<=150;i+=30){
      Angle_detection(i);
  }
  Angle_detection(175);
}

/*****************************其余功能函数*********************************/
/*
*@name:keysacn
*@function:按键启动
*/
void keysacn()
{
  int val;
  val=digitalRead(key);                         //读取数字 7 口电平值赋给 val
  while(!digitalRead(key))                      //当按键没被按下时，一直循环
    ;
```

```
  while(digitalRead(key))//当按键被按下时
  {
    delay(10);                                    //延时 10ms
    val=digitalRead(key);                         //读取数字 7 口电平值赋给 val
    if(val==HIGH)                                 //第二次判断按键是否被按下
      digitalWrite(beep,HIGH);                   //蜂鸣器响
      if(!digitalRead(key))                     //判断按键是否被松开
        digitalWrite(beep,LOW);                  //蜂鸣器停止
    else
      digitalWrite(beep,LOW);                     //蜂鸣器停止
  }
}
/*****************************loop 主函数**********************************/
void loop() {
  keysacn();                                     //按键启动
  int SR_2,SL_2;
  while(1){
    SR_2=digitalRead(SensorRight_2);
    SL_2=digitalRead(SensorLeft_2);
    front_detection();//测量前方距离
    //前方存在障碍物
    if((SR_2 == LOW || SL_2 == LOW)||(Front_Distance < 20)){
      digitalWrite(beep,HIGH);              //蜂鸣器响
      Serial.print("D");
      brake();
      delay(300);
      while(1){
          getstr=Serial.read();             //直到读到后退才能退出
          if(getstr=='B'){
            Serial.print("b");
            back();
            delay(1000);
            digitalWrite(beep,LOW);         //蜂鸣器停
            break;
          }
      }
    }
    else
      digitalWrite(beep,LOW);               //蜂鸣器停
    //接受蓝牙串口信息
    getstr=Serial.read();                  //读取蓝牙串口获取的信息
    /*
    *F:前进
    *S:停下
```

```
    *B:后退
    *L:左转
    *R:右转
    *W:超声波扫描
    */
    if(getstr=='S'){
      Serial.print("s");
      brake();
    }
    else if(getstr=='F'){
      Serial.print("f");
      run();
    }
    else if(getstr=='B'){
      Serial.print("b");
      back();
    }
    else if(getstr=='L'){
      Serial.print("l");
      spin_left();
    }
    else if(getstr=='R'){
      Serial.print("r");
      spin_right();
    }
    else if(getstr=='W'){
      Serial.print("i");                //watch start
      delay(20);
      Ultrasonic_Detection();           //进行角度扫描
      delay(20);
      Serial.print("e");                //watch end
    }
    else                                //无指令 waiting
        ;
  }
}
```

五、 测试模块建模

1.uart 蓝牙串口模块 testbench

```
`timescale 1ns / 1ps
module uart_run_tb();

reg clrn;
```

```
reg clk;
reg [7:0] d_in;
reg write;
reg rxd;

wire[7:0] d_out;
wire txd;
wire get;
uart_run #(0)uart(.clrn(clrn),.clk(clk),.d_in(d_in),.d_out(d_out),.write(write),.rxd(rxd),.txd(txd),.get(get));

reg [7:0] invect;
reg [7:0]cou1;
reg [7:0]cou2;
initial
begin
	cou1<=0;
	cou2<=0;
	write<=1'b1;
	clk<=1'b0;
	d_in<=8'b01101110;
	clrn=1'b0;
	#10
	rxd=1'b1;
	for (invect = 0; invect < 9999999; invect = invect + 1)
		begin
		cou1=cou1+8'b00000001;
		write=~write;
		clrn=1'b1;
		#10
		clk=~clk;//time keeper
		if(cou1>=16)
		begin
			rxd<=1'b1;
			cou2<=cou2+8'b00000001;
			if(cou2>=14)begin
				rxd<=1'b0;
				cou2<=8'b00000000;
			end
			cou1<=1'b0;
		end
		end
end

endmodule
```

2. vga 模块 testbench

```
`timescale 1ns / 1ps
module VGAtest_tb;
	reg I_clk;
	reg I_rst_n;
	wire [3:0] O_red;
	wire [3:0] O_green;
	wire [3:0] O_blue;
	wire O_hs;
	wire O_vs;
	reg [7:0] invect;
vga_driver
vga(.I_clk(I_clk),.I_rst_n(I_rst_n),.O_red(O_red),.O_green(O_green),.O_blue(O_blue),.O_hs(O_hs),.O_vs(O_vs))
;
initial
begin
		I_clk<=1'b1;
		I_rst_n=1'b0;	//先置位!!!
		#10
		for (invect = 0; invect < 9999999; invect = invect + 1)
			begin
			I_rst_n=1'b1;
			#10
			I_clk<=~I_clk;
			end
end
endmodule
```

3.car_top 模块 testbench

```
`timescale 1ns / 1ps
module car_tb();
reg clk;
reg I_rst_n;
reg rxd;
reg go;
reg excute;
reg [4:0]setdir;
wire txd;
wire [3:0] O_red;
wire [3:0] O_green;
wire [3:0] O_blue;
wire O_hs;
wire O_vs;
wire [4:0]showstate;
```

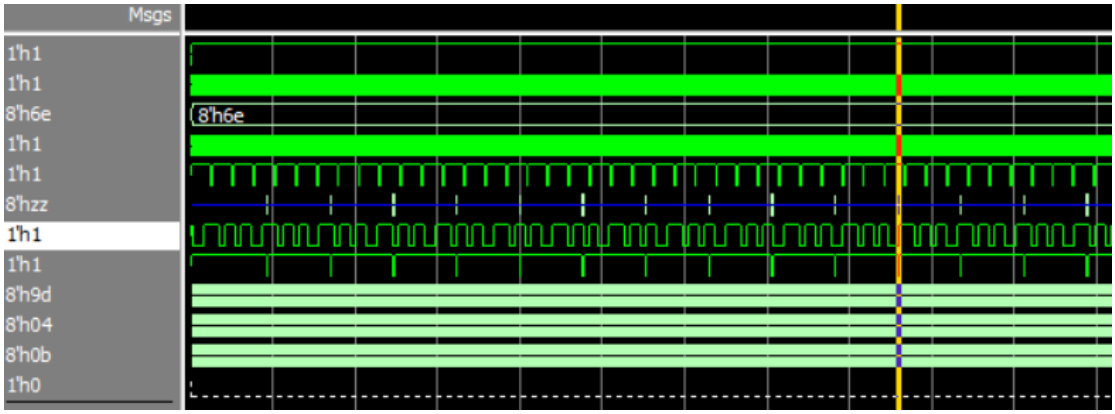
```
car_top car(
        .clk(clk)           ,
        .I_rst_n(I_rst_n)   ,     // low level effective
        .rxd(rxd)           ,
        .go(go)             ,     // high level effective
        .excute(excute)     ,     // high level effective
        .setdir(setdir)     ,     // for test traxial...
        .txd(txd)           ,
        .O_red(O_red)       ,     // VGA's red     channel
        .O_green(O_green)   ,     // VGA's green channel
        .O_blue(O_blue)     ,     // VGA's blue    channel
        .O_hs(O_hs)         ,     // VGA
        .O_vs(O_vs)         ,     // VGA
        .showstate(showstate)       // show for test
        );
reg [7:0] invect;
initial
        begin
                I_rst_n<=1'b0;
                clk<=1'b0;
                rxd<=1'b1;
                go<=1'b1;
                excute<=1'b1;
                setdir<=5'b00000;
                #10
                for (invect = 0; invect < 9999999; invect = invect + 1)
                                begin
                                I_rst_n<=1'b1;
                                #10
                                clk=~clk;//time keeper
                                end
        end
endmodule
```

其余模块无需 testbench，直接下板效果更明显。

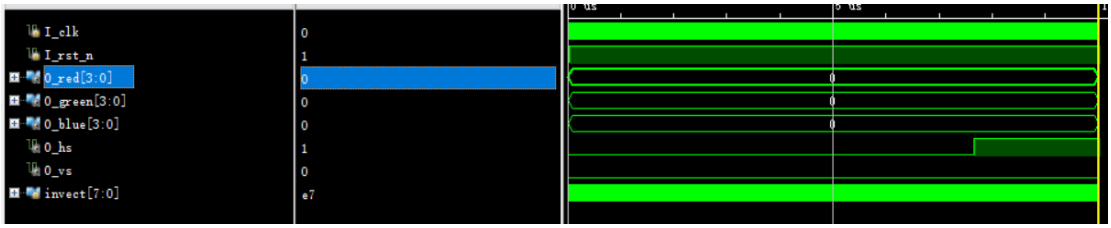
六、 实验结果

1.ModelSim 仿真图

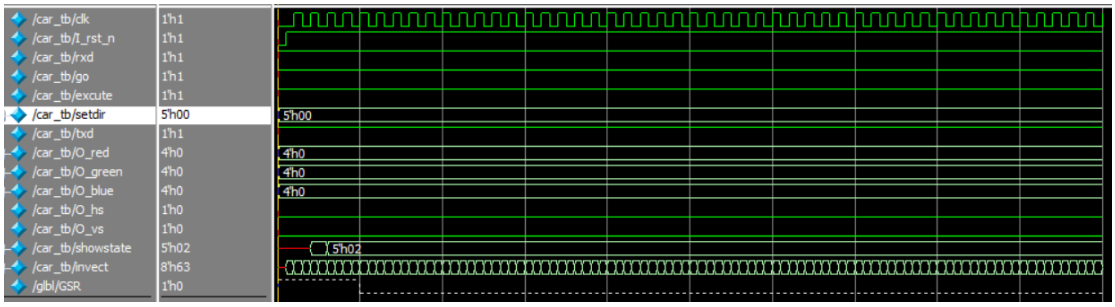
uart 蓝牙串口模块仿真图：



vga 模块仿真图：



car_top 模块仿真图：

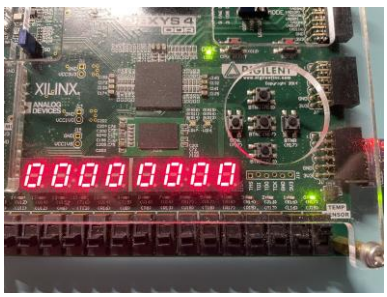


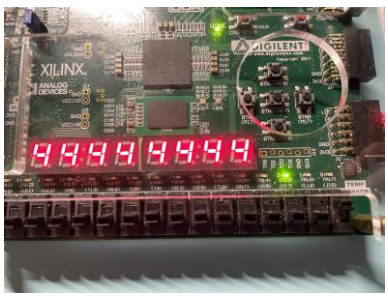
2. 下板实验结果贴图

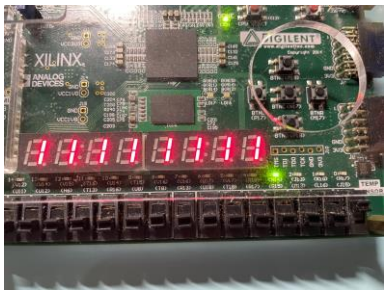
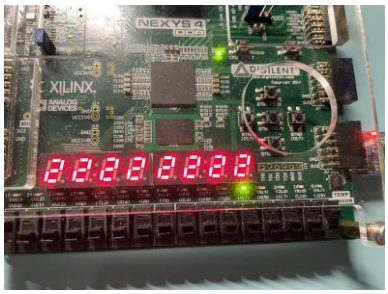
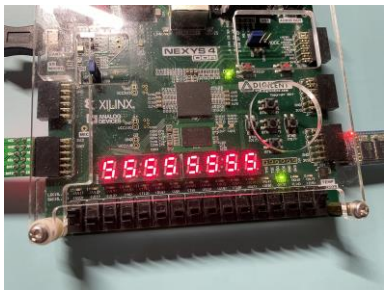
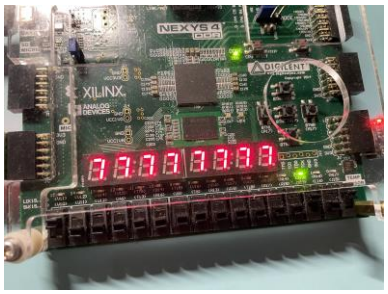
1.蓝牙信息接收情况：

1	2	3	4	5	6	7	8
D	i	e	s	f	l	r	b
遇障信号	探测开始	探测结束	停止	前进	左转	右转	后退

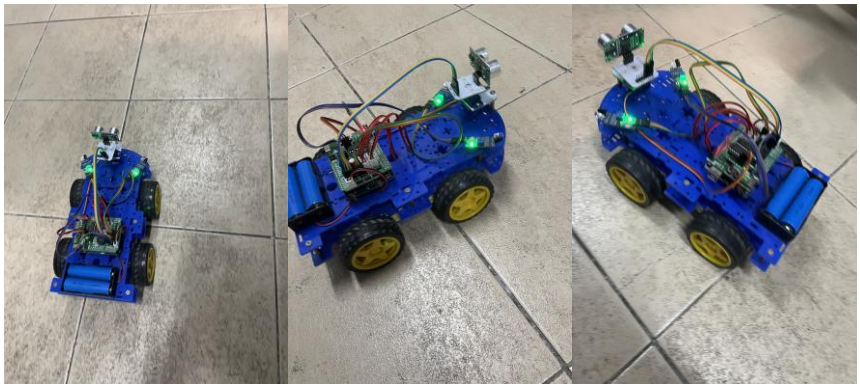
R18	N14	J13	K15	H17
SEARCH	WARNING	RUN	INTER	START



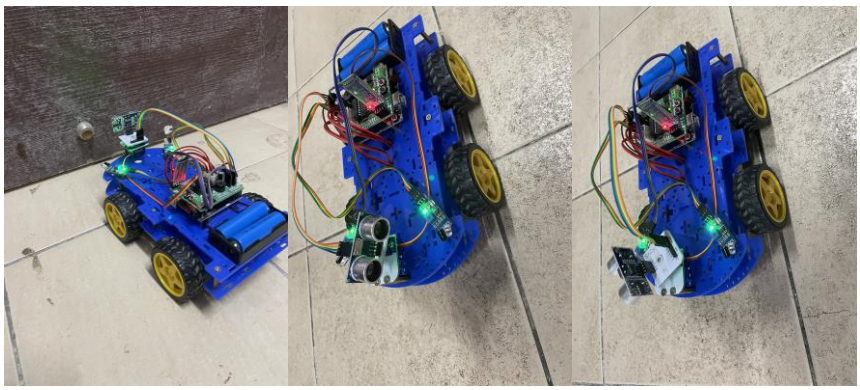


2.小车行驶情况



3.小车环境感知情况



4.VGA 实时定位

七、 结论

无论是 uart 模块、vga 模块、triaxial 模块抑或是 arduino 模块都能较好地实现功能。组合起来也能在控制器的调度下相互协调控制，实现既定功能。因此，本项目总体来说实现了预期：slep 能初步实现可操纵的实时定位与环境感知，进而实现对陌生环境的地图构建。

但是也有很多值得改进的地方。L3G4200D 三轴传感器的传感精度和及时性不好，无法在小车的操纵过程中实现较好的及时控制。可能是对于三轴传感器的了解还不够深入导致这一原因。因此，对于 L3G4200D 的最优读取速率，高通、低通滤波 器，Interrupt 功能等的使用仍然可以进一步探索，获取到更加优质的数据。

基于 slep 还有很多值得提升的地方：在小车的操纵过程中可以使用更方便的外围部件，优化操作过程；可以使用摄像头代替超声波雷达，实现更好的外部环境感知；可以构建自主探测系统，当小车遇障后可自行解决，重回正常，进而实现自主探测。

八、 心得体会及建议

在本项目的实现过程中，遇到了较多困难。在硬件设施的使用上常出现较多意想不到的错误，并且调试较为困难。例如最开始如何将两个主从蓝牙模块进行连接就花费了较长时间，因为只有使用串口调试工具改变蓝牙部件的状态，然后让其自行配对，对其配对过程无法做到像使用手机做主模块那样一直操控。配对的未知性便是一大挑战。无论是超声波雷达舵机的 PWM 波、L3G4200D 的 SPI 通信协议还是 VGA 的泰勒展开三角函数都是实现过程中遇到重重障碍。不过也正是这些困难，才使得 slep 的成功实现具有极大意义。

通过数字逻辑本课程的学习，从最基础的门电路，到组合逻辑比较器，再到寄存器堆，都让我对于芯片最底层的具体构造有了更加清晰地认识。亲手实现每一个基本元器件，了解元器件背后的构成原理，是我在这门课上的最大收获。当然，这门课程也极大的提升了我的问题解决和稳定心态的能力。硬件设备极难查错，在调试过程中也极易出现各式各样的错误。因此，在本门课程做实验的过程中，常常被各种 bug 所缠绕。面对错误，冷静处理，检查逻辑，稳定心态是这门课教会我的处理办法。

本课程总体来说：课程设计严谨、实验选择恰当、老师认真负责。是一门较好的计算机硬件课程。但是在也有一些值得提出的建议。老师授课过程可以增加一些实际操作，或者更改以下 ppt 内容排布。课上仅通过鼠标指点，对知识进行生硬传授，往往授课效果不佳。课下需要花费大量时间学习本应该在课上学到的知识，这样既浪费了时间，也使得老师的努力白费。实验课上，同学们往往会遇到很多困难，老师应当在实验课开始前让同学们自行摸索，提升同学的实际操作能力。但希望在实验课结束时，能对本实验的困难点给出提示或相应知识背景介绍，避免同学们在课下花费大量时间而收效甚微。

习近平总书记曾说：核心技术是要不来的。目前芯片产业已俨然成为我国的一项卡脖子技术。目前，国内半导体自给率水平非常低，特别是核心芯片极度缺乏，国产占有率几乎为 0，芯片关乎国家安全，国产化迫在眉睫。因此，作为当代高校学生，是先进技术的生力军，是国家民族未来的核心力量。我们应当时刻关注我国所面临的尖端技术瓶颈，投身到尖端科研技术的研发过程中，为实现科技强国的宏伟目标贡献自己的一份力。

诚如黄大年同志所说：“若能做一朵小小的浪花奔腾，呼啸地加入时代献身者的滚滚洪流中推动历史向前发展，这将是一生中最值得骄傲和自豪的事情。”

与诸君共勉。