一辆真正的汽车!

小燕拿到了驾照，很想拥有一辆高科技的真车。有一天，小燕买了一辆玩具车，它真的很容易使用。它只有4个操作：直行、倒车、左转、右转。....."嘿，等等。我的车不可能是这种垃圾。"

为了体验驾驶真车的感觉，Yan想改变汽车的运行方式，他向你寻求帮助。因此，你把它连接到EGO1开发板。你需要开发一个程序，将真车的 "复杂操作 "转换为小玩具车的4个简单操作。

另外，由于严某想要一辆高科技的真车，所以希望你能开发手动驾驶模式、半自动驾驶模式和自动驾驶模式。

虽然这只是一辆玩具车，但它却承载着小燕的幻想梦想。你能帮助他吗？[TOC]

基础：全球状态（20%）。

汽车有2种状态：**通电**和**断电**。我们将使用2个按钮来管理基本状态的切换。

打开电源（按钮）

按住开机键至少1秒，汽车将启用其发动机。如果汽车的发动机没有启用，所有其他的输

入就无法使用。

关闭电源（按钮）

按下电源关闭按钮，汽车将禁用其发动机。在后面的介绍中，有一些特殊情况会导致发动机失效。

当汽车通电时，它最多有3种模式可以选择。

手动驾驶模式、半自动驾驶模式、自动驾驶模式（备选）。

然后我们将分别介绍这三种模式。

基础：手动驾驶（50%）。

在手动驾驶模式下，该车可以作为真正的汽车来操作。

国家分析

在手动驾驶模式下，汽车有3种状态。

不启动，启动，移动。

最初（当汽车启用其发动机，或切换到手动模式），汽车应该处于不启动状态。当且仅当它处于移动状态时，汽车才能移动。

输入

节流阀（开关）

如果汽车被启动，你可以直接使用油门来控制汽车是否应该移动。

但如果汽车没有启动，你需要用油门和离合器来启动汽车，否则汽车可能会失去动力！"。

> 当打开油门的时候。
> 如果汽车处于启动状态，则切换到移动状态。
> 如果汽车处于不启动状态，我们应该检查离合器。
>
> 关掉油门时。
> 如果汽车处于移动状态，则切换到启动状态。

离合器(开关)

每当离合器被启用时，汽车就不能移动。在汽车没有启动的情况下，我们需要离合器来使汽车启动。我们应该首先保持离合器开启，然后打开油门，完成汽车的启动。这样做之后，我们就完成了启动。

如果我们在汽车需要启动时直接打开油门（不打开离合器），汽车发动机将停止工作，需要重新通电。

> 当汽车处于不启动状态时。
> 打开油门**，启用离合器**，切换到启动状态。 打开油门**，不启用离合器**，汽车会断电
> 。
>
> 当汽车处于启动/行驶状态时。
> 如果离合器打开，汽车必须离开移动状态，并且不能切换到移动状态。

刹车(开关)

打开刹车开关，让汽车停下来。刹车后，汽车需要再次启动。

请注意，大多数汽车都有刹车超控系统（BOS）。如果我们同时打开刹车和油门，刹车应该工作，而油门不应该工作。Yan的车也应该有BOS。

> 当汽车处于启动/行驶状态时。
> 打开刹车，切换到不启动状态。如果刹车打开，油门开关就无
>
> 效了。

逆向换挡（开关）

如果这个开关是打开的，汽车应该向后移动而不是向前移动。

切换这个开关时，离合器必须持续打开，否则汽车将失去动力。

> 当汽车处于移动状态时。
> 如果倒档开启，汽车应该向后移动。
> 如果倒档关闭，汽车应该向前移动。
> 如果我们在没有启用离合器的情况下，在移动状态下切换倒档，汽车会断电。

左转和右转（按钮）

> 如果汽车处于启动状态或移动状态。
> 当只按下左转按钮时，汽车应该向左转。当只按下右转按钮时，汽车应该向右转
> 。
> 否则（没有人按下或两个人都按下），汽车应该直线行驶。

输出

汽车操作（对UART）。

你可以通过激活uart模块中的信号来控制汽车。总共有4个信号：直行、返回、左转、右转。

更多细节，请到UART模块介绍中查看。

转向灯（LED）

有两个LED作为转向灯：如果汽车没有启动，这两个LED应该一直亮着。否则，如果汽车正在左转，左边的LED应该闪烁。

否则，如果汽车正在右转，右边的LED应该闪烁。

里程记录（7段）

当汽车通电后，里程记录需要显示在七个数码管中。当汽车处于移动状态时，里程数应不断增加。

当汽车断电时，里程数应重置为0，并且不显示任何数值（可能是"-"或只是没有显示）。

高级。半自动驾驶 (30%)

在半自动驾驶模式下，汽车会自动直行，在岔路口停下，然后等待我们的指令（直行或左转或右转）。

国家分析

该车在半自动驾驶模式下有3~4种状态。移动，转弯（左转和右转），等待指令。

最初，汽车处于移动状态。当汽车走到一个岔路口时，它将切换到等待命令状态。发送命令后，汽车将根据命令切换到转弯状态或移动状态。

输入

检测器（来自UART）

在汽车的每个方向有4个探测器。你可以用它们来探测汽车附近是否有障碍物。

半自动驾驶指令（多按钮）。

汽车会自动走直路。我们可以用按钮来让汽车知道它是否应该直行、左转、右转或在岔路口返回。

> 在移动/转弯状态下。
> 该命令按钮无效。
>
> 处于等待命令状态。
> 3个按钮可以切换成3种相应的状态：左转、右转、直行。
> 当汽车转过90度时，转弯状态将自动切换为移动状态。

提示

1. 你应该使用4个探测器的信息来确定汽车是否在岔路口。
2. 在转弯状态下，你需要估计转90度的时间。转弯状态只能根据**时间**自动切换。提供一个50Hz的时钟是一个不错的选择。
3. 4个检测器的数值可能不会同时变化（时间差约为40ms）。不要让检测器成为你 "永远 "的触发信号。
4. 建议在转弯和直行之间增加一个冷却时间。因为你的车需要很短的时间来自动校准方向。

奖励：自动驾驶（额外20%）。

如果汽车处于自动驾驶模式，那么汽车应该自动开出一个迷宫，不需要额外的操作!

输入

放置信标（到UART）。

在这个信号的情况下，汽车会在原地放置一个信标。该信标也会被你的汽车探测器探测到（就像一个障碍物），但汽车可以穿过信标。

在地图上最多可以有10个信标同时存在。如果有10个以上的信标，第一个放置的信标将被摧毁。(警告：这个信号必须保持活跃超过20ms，否则会因为缓冲机制而无效)。

销毁信标（至UART）

在这个信号的情况下，汽车将摧毁最近放置的信标。(警告：这个信号必须保持活跃超过20ms，否则会因为缓冲机制而无效。)

提示

1. 右转第一原则对走出迷宫很有帮助，但要小心。迷宫中的一个循环会使这一原则失效。
2. 我们为测试自动驾驶准备的迷宫很简单。你的车需要在有限的时间内到达目的地。
3. 信标对于检测循环和穿越迷宫非常有帮助。
4. 即使你的车没有到达目的地，只要它是完全自主的，不撞墙，你就可以得到保证的奖励分。

奖金：VGA（额外的20%）。

任务1。可用的VGA

使用开关来控制VGA显示任何东西。

任务2.使用VGA来显示状态

让VGA显示你的车的状态。

任务3：使用VGA显示里程记录

让VGA显示里程记录。(实时同步)

附录。UART模块的介绍

```
module SimulatedDevice(
    input sys_clk, //system clock (100Hz, P17 pin)
    input rx, //bind to N5 pin
    output tx, //bind to T4 pin

    input turn_left_signal,
    input turn_right_signal,
    input move_forward_signal,
    input move_backward_signal,
    input place_barrier_signal,
    input destroy_barrier_signal,
    output front_detector,
    output back_detector,
    output left_detector,
    output right_detector
    );
```

该模块声明如下。

系统接口

sys_clk,rx,tx需要直接访问引脚。

```
sys_clk --------- P17
rx ---------------- N5
tx ---------------- T4
```

输入接口

6个输入信号允许你操作汽车： **turn_left_signal** 如果这个信号激活，汽车将继续向左转。 **turn_right_signal** 如果这个信号激活，汽车将继续向右转。 **move_forward_signal** 如果这个信号激活，汽车将继续向前移动。 **move_backward_signal** 如果这个信号激活，汽车将继续向后移动。

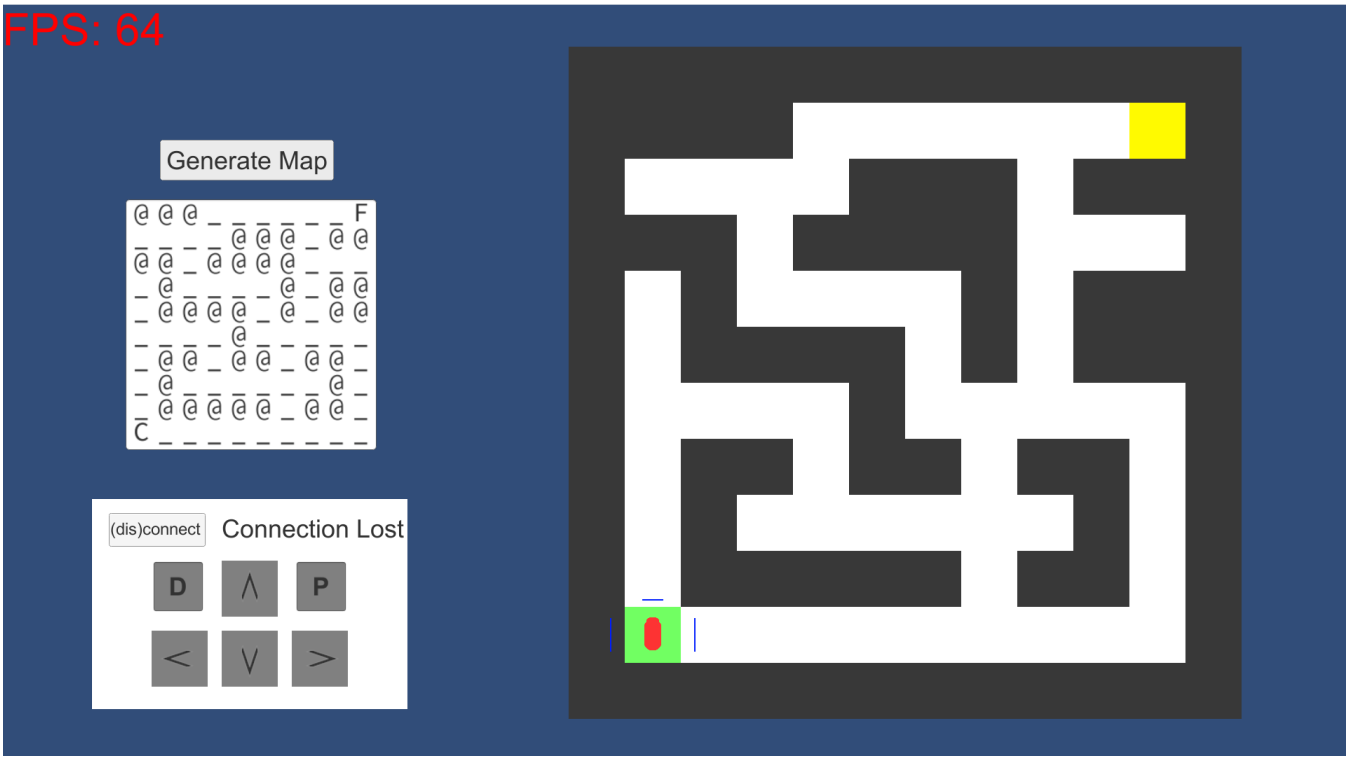
place_barrier_signal 该车将在这个信号的位置放置一个信标。(警告：这个信号必须保持激活超过20ms，否则会因为缓冲机制而无效) **destroy_barrier_signal** 如果这个信号是激活的，汽车会摧毁当前放置的最多的信标。(警告：这个信号必须保持活跃超过20ms，否则会因为缓冲机制而无效)

输出接口

4个输出信号是来自检测器的信息。 **front_detector** 如果汽车前面的检测器发现有墙/becon（换句话说，汽车前面有墙），这个信号将被激活。 **back_detector** 如果后面有墙，这个信号将被激活。 **left_detector** 如果左边有

墙，这个信号将被激活。 **right_detector** 如果右边有墙，这个信号将被激活。

附录。仿真软件介绍



整个软件有三个部分：地图生成器、控制器、迷宫。

地图生成器

我们用4个字符代表迷宫中的4类物体。**@**是迷宫中的墙。_是迷宫中的道路。**C是迷宫中的**汽车。**F是迷宫中的**目的地。

控制器

它负责通过UART连接开发板并接收控制信号。如果控制器从你的开发板上获得控制信号，你可以看到相应的按钮高亮显示。

迷宫

你可以在迷宫中看到你的车（红色）。你的目标是把车开到目的地（黄色）。

你可能会看到汽车周围有四条蓝线，它们是检测器的有效工作范围。

附录。如何开始？

资源清单

我们有6种资源。

1. **CarSimulation** 该软件提供了一个模拟汽车和一个迷宫。
2. **SimulatedDevice_demo.bit** 这是模拟设备的演示， 可以直接运行。
3. **SimulatedDevice_src** 演示的源代码。

4. **State_reference_diagram.pdf** 状态转换供参考。仅供参考！！
5. **DL2022F_Project_Introduction.pdf**就是这个文件。
6. **semi_auto_driving_mode_demo.bit 半自动驾驶**模式的演示。

第1步：尝试SimulatedDevice演示

要使用这个演示，将**SimulatedDevice_demo.bit**编程到你的开发板上，然后运行CarSimulation软件。如果连接成功，那么你可以使用从左到右的前六个开关来控制汽车。同时，四个LED灯从右到左计数，显示来自检测器的反馈!如果连接失败，请检查开发板是否正确连接和编程，然后点击软件中的按钮进行重新连接。

第二步，将SimulatedDevice导入你的项目中

演示中的所有源代码（文件夹**SimulatedDevice_src**）构成了与汽车互动的设备。这个设备的主要部分是一个UART模块。你只需要把所有的源代码文件添加到你自己的项目中。

你需要在你的项目中实例化的模块在文件**dev_top.v**中被称为**SimulatedDevice**，我们之前刚刚介绍了带有UART模块的设备。

不要修改演示中的任何代码，并确保**SimulatedDevice**只被实例化一次。

第三步。设计你自己的模块

完成前两个步骤后，你可以开始设计和实现你的模块了。**State_referece_diagram**可以帮助你理解状态之间的关系，但请记住，这个图只供参考，不作要求！！。

第四步：尝试半自动驾驶演示

由于对半自动驾驶模式的介绍可能不够详细，我们还提供了一个半自动驾驶模式的演示。要使用该演示，首先要编写**Semi_auto_driving_mode_demo.bit**，然后运行CarSimulation软件。

你可以使用前4个开关从左到右来控制汽车。

- **P5引脚**走直线的命令。
- **P4引脚**向左转的指令。**P3引**
- **脚**向右转指令。**P2引脚**向后
- 转指令。