9032 project report

Z5196480

Huiyao zuo

1. **the general description about the project development, management, and the contributions of each group member**

   in the project development part .we first analysis the project problem and discuss the structure of code , and decided this project mainly has three part , interrupt and time counter , output ,logic part. then we discuss the logic flow together.

   In the project management part . after we discuss the project content ,we have write code separately and because other member has already go to the lab3 and lab4 ,so show board part is my responsibility in the project. since one member write a great code ,so we decided use this version' s code and I do the test part.

   the code is mainly complete by other two members , and I responsible for test the code in the board and find the bug and fix the code .but I have also write a version of code ,but this version need more register than the code we use now ,and I will stick my code in the last part.

2. **the overview of the project design, which includes:**

hardware components used and related interfacing design:

hardware components includes the interrupt 0 & 1 , one time counter ,LED ,LCD ,keypad ,port f ,port c ,port a ,port d. pb0 & pb1.

Pb0 linked to interrupt 0 , when pb0 be pushed ,the voltage change and trigger the interrupt 0.

Pb1 linked to interrupt 1, when pb1 be pushed ,the voltage change and trigger the interrupt 1

LED link to port c , port c as the output to LED.

LCD link to port f and port a. port f output the data and port a output the commend.

Keypad link to port f, and f read the information of keypad.

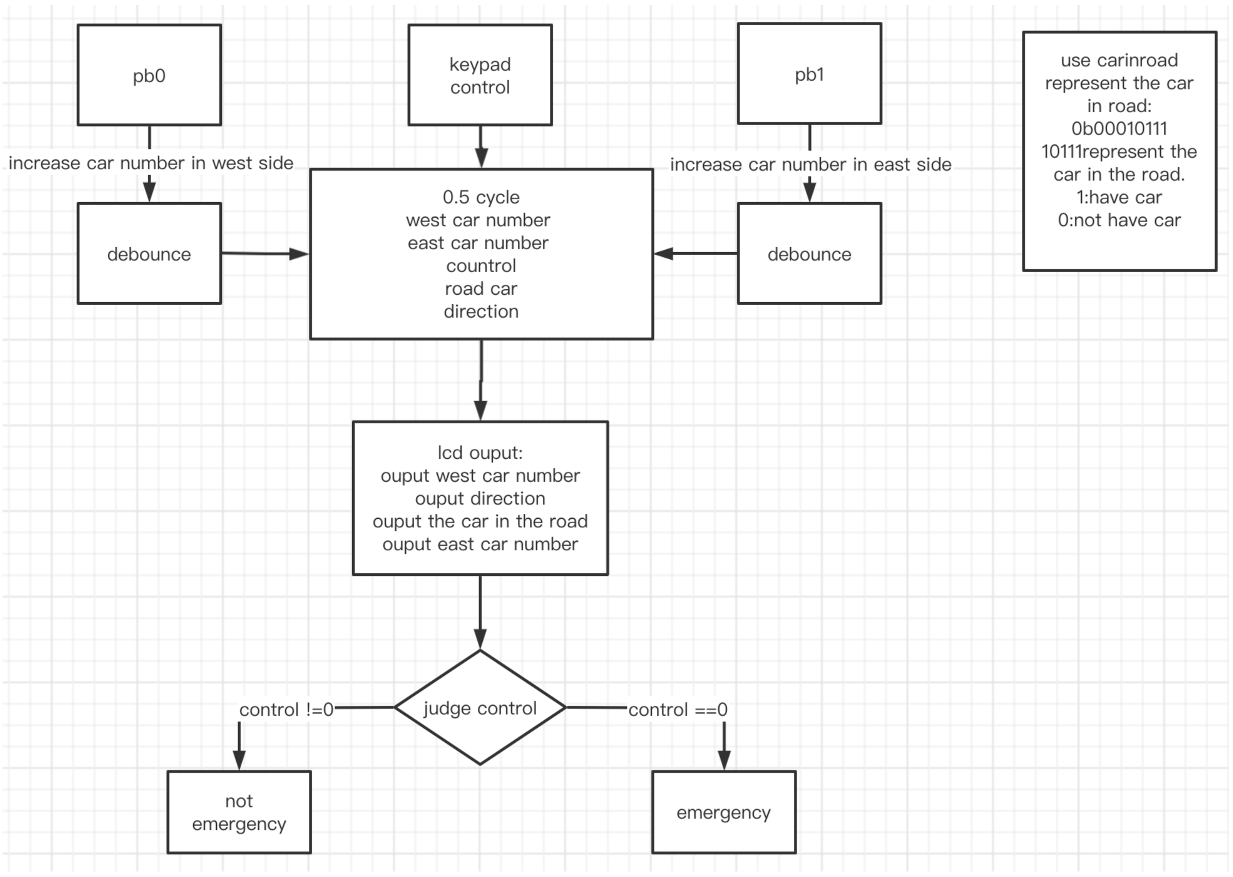
software code structure and execution flow

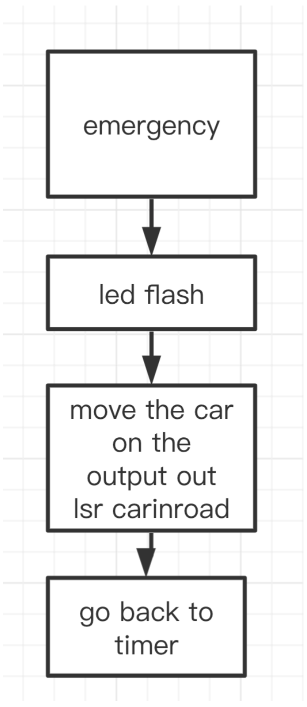
software includes three part:

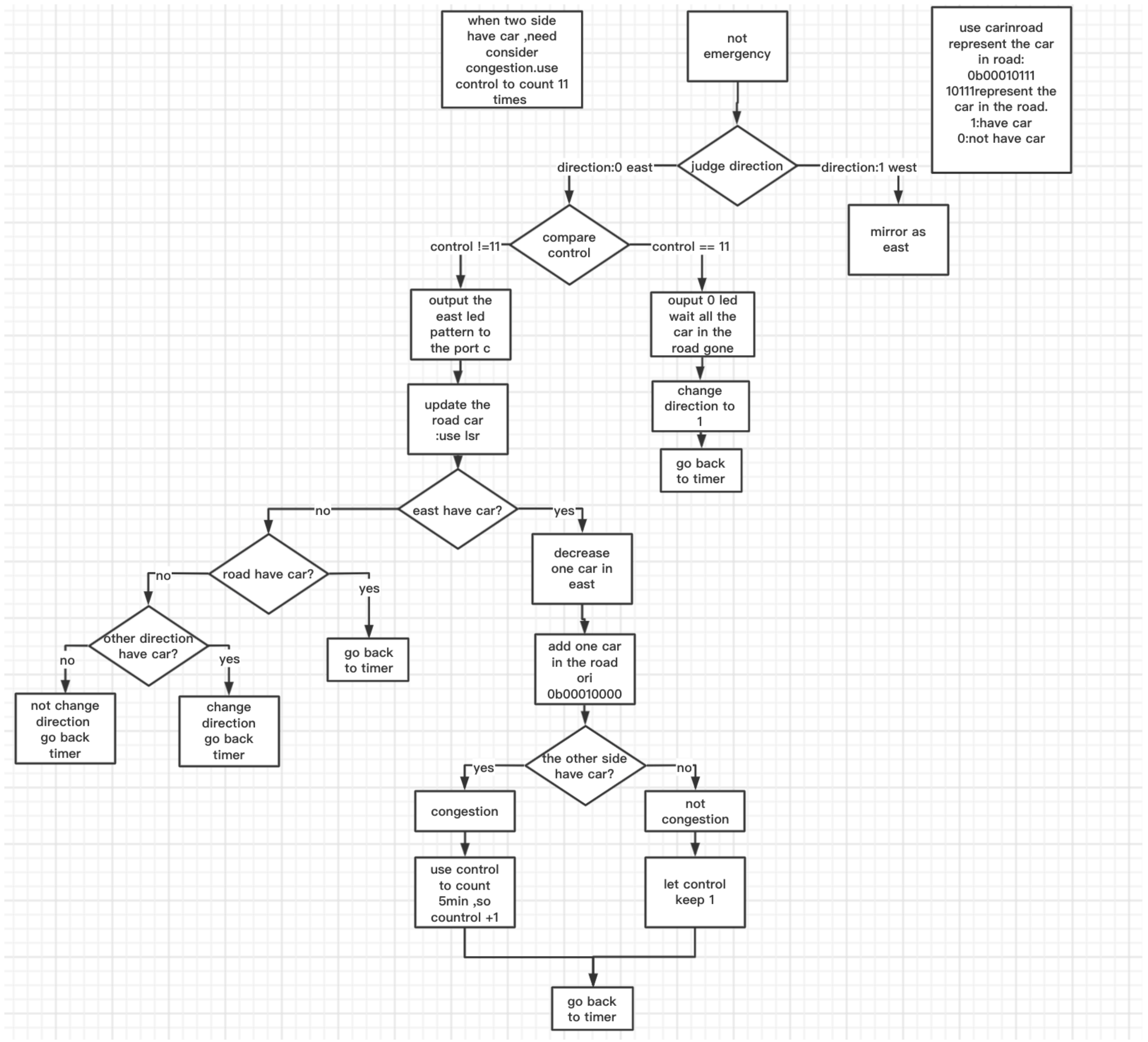
1. LCD output part
2. logic function part

3. timer and interrupt part

The flow picture are followed.


pb0
keypad control
pb1
use carinroad represent the car in road: 0b00010111 10111represent the car in the road. 1:have car 0:not have car
increase car number in west side
increase car number in east side
debounce
0.5 cycle west car number east car number countrol road car direction
debounce
lcd ouput: ouput west car number ouput direction ouput the car in the road ouput east car number
judge control
control !=0
control ==0
not emergency
emergency





when two side have car ,need consider congestion.use control to count 11 times
not emergency
use carinroad represent the car in road: 0b00010111 10111represent the car in the road. 1:have car 0:not have car
judge direction
direction:0 east
direction:1 west
mirror as east
compare control
control !=11
control == 11
output the east led pattern to the port c
ouput 0 led wait all the car in the road gone
change direction to 1
go back to timer
update the road car :use lsr
east have car?
no
yes
road have car?
decrease one car in east
no
yes
other direction have car?
go back to timer
add one car in the road ori 0b00010000
no
yes
not change direction go back timer
change direction go back timer
the other side have car?
yes
no
congestion
not congestion
use control to count 5min ,so countrol +1
let control keep 1
go back to timer

how software and hardware interact with each other.

a. pb0 Butten trigger the interrupt 0 add the car numbers in west side (this is a separate part in the code and include debouncing preprocessing.)
b. pb1 Butten trigger the interrupt 1 add the car numbers in east side (this is a separate part in the code and include debouncing preprocessing.)
c. port F used to read keypad value (before and after the LCD output,because use the port F as LCD output and keypad input at same time)
d. port F used to output data value to LCD
e. port A used to output commend value to LCD
f. port C used to output the LED pattern
   this system is fresh every 0.5s. during one 0.5s cycle, the interrupt would increase the car number in each side if the pb0 or pb1 be pushed. and the LCD output part would output the car num each part side and the road in the road, and the direction.then the logic part would update the car number, direction ,and car in road.

3. conclusive remarks about the project.

There are many things need attention in this project. How to solve the problem clever is very important. For the push debouncing problem , when set the value cap between the push ,if the value set too small ,there would still have great raising in the car number ,but when the cap set too large ,when push the button and want to add car numbers would very difficult. I was use number to count the car number in the road ,but this would need more register. and in the output car number part ,when the num is over 99,the output would have something wrong ,but the car number is still right ,that is because the output marcro only consider maximum two tigit number. And some problem in the code is very interesting, in the emergency judge ,the original code is dot use breq to judge directly, because the emergency is too far to jump ,then add a middle in the code ,so like jump two times to the emergency. and the output part also need consider carefully. In conclusion ,the output part and hardware part need take carefully and the soft logic part need more logical thinking, and this project really help me to understand more of avr.

4.my original code , not use the 0b00011111 to count the code in the road ,so is more complex than the code we use now.and I would only stick the one side and only the logic part. And because this would use more register so it would make it more hard to combine software and hardware.

```
stopinside_judge:     ;this is the decide which situation is
```

```
    cpi carnum0 ,0
    breq car0_0_judge
    cpi carnum1 ,0
    breq only_direction0_have_car
    rjmp shuangbian_prepocess

car0_0_judge:
    cpi carnum1 ,0
    breq road_car_change
    rjmp only_direction1_have_car

road_car_change:
    cpi roadnum , 0
    breq output0
    rjmp wait_car_leave
output0:
    jmp output



only_direction0_have_car:    ;this is only one side have car ,but the road have
                               other side car judge
    cpi roaddirection , 0
    breq danbian0
    rjmp wait_other_direction_car0

wait_other_direction_car0:
    cpi roadnum , 0
    breq danbian0
    rjmp wait_car_leave



wait_car_leave:
    inc timercap
    cpi timer,0
    breq wait_car_leave_then
    dec timer
    rjmp output
wait_car_leave_then:
    dec roadnum
    rjmp output



danbian0:                ;this is only one side have car
```

```
    ldi currentdirection ,0
    cpi roaddirection , 0
    brge resettimer0
    cpi timer , 0
    breq capjudge0
    rjmp danbianprocess0
resettimer0:
    ldi timer, 4
    ldi roaddirection,0
    ldi timercap ,0
    rjmp danbianprocess0
capjudge0:
    cpi timercap ,0
    breq danbianprocess0
    mov timer, timercap
    ldi timercap ,0
    rjmp danbianprocess0


danbianprocess0:
    cpi roadnum,5
    brlo danbian2
    rjmp danbian4
danbian2:
    inc roadnum
danbian4:
    cpi carnum0, 0
    brge danbian6
danbian6:
    dec carnum0
    dec timer
    jmp output

shuangbian_prepocess:            ;this is consider two side have car
    cpi count7, 1
    brlo shuangbianchangedirection
    cpi currentdirection , 0
    breq shuangbiandirection0
    rjmp shuangbiandirection1

shuangbianchangedirection:
    cpi currentdirection , 0
    breq cahngeto1
    ldi count7 , 7
    ldi currentdirection ,0
    rjmp shuangbiandirection0


cahngeto1:
    ldi currentdirection ,1
    ldi count7 , 7
    rjmp shuangbiandirection1

shuangbiandirection0:
    cpi roaddirection , 0
    brge wait_car_leave_shuang
    rjmp danbian_shuang0

wait_car_leave_shuang:
    inc timercap
    cpi timer,0
    breq wait_car_leave_shuang_then
    dec timer
    rjmp output
wait_car_leave_shuang_then:
    dec roadnum
    rjmp output
;

danbian_shuang0:
    cpi roaddirection, 0
    brge resettimer_shuang0
    cpi timer , 0
    breq capjudge_shuang0
    rjmp danbianprocess_shuang0
resettimer_shuang0 :
    ldi timer, 4
    ldi roaddirection,0
    ldi timercap ,0
    rjmp danbianprocess_shuang0
capjudge_shuang0:
    cpi timercap ,0
    breq danbianprocess_shuang0
    mov timer, timercap
    ldi timercap ,0
    rjmp danbianprocess_shuang0

danbianprocess_shuang0:
    cpi roadnum,5
    brlo danbian_shuang2
    rjmp danbian_shuang4
danbian_shuang2:
    inc roadnum
danbian_shuang4:
    cpi carnum0, 0
    brge danbian_shuang6
danbian_shuang6:
    dec carnum0
    dec timer
    jmp output
```