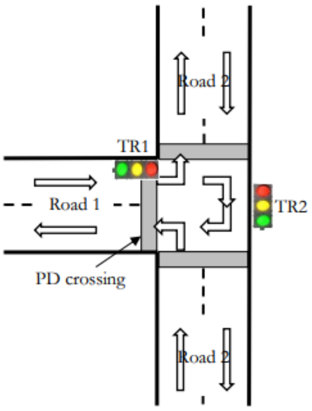
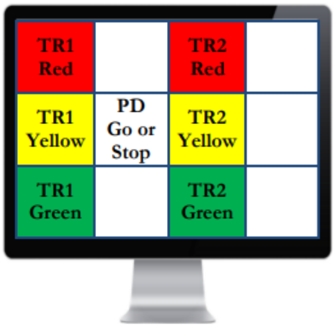
# Guidebook of Traffic Light

## I.Basic

### Introduction and description of the application and its functionalities:

This is a project of traffic light system for a T-junction, which comprises of two traffic lights and a pedestrian light. There are nine regions on the monitor using to represent the traffic light system. The regions in the first column represent for the red, yellow and green light for the traffic light 1 and the regions in the third column represent for the traffic 2. The region in the center represents for the pedestrian light. The map and region are shown as bellow.





The road light operates in that way: 1R2R -> 1R1Y2R -> 1G2R ->1Y2R -> 1R2R ->1R2R2Y ->1R2G->1R2Y->1R2R .....next period. If the pedestrian button is pressed, how the program will operate will contingent on the moment of pushing. If it is pushed when two traffic light is right, after the second traffic light return from yellow, the pedestrian light will turn green for five second. If is pushed at other time in the period, the pedestrian light will turn green after the finish of a normal period(traffic light two turn red from yellow). In the five second of pedestrian light turn green, the first three second it will turn up constantly, in the last two second it will blink for five times. After the pedestrian light return to red, it will divert to 1R1Y2R mode. What's more, how many time of this mode is left will show on the seven segment decoder. And the same pattern will show on the LEDs light. That is the basic function of the program.

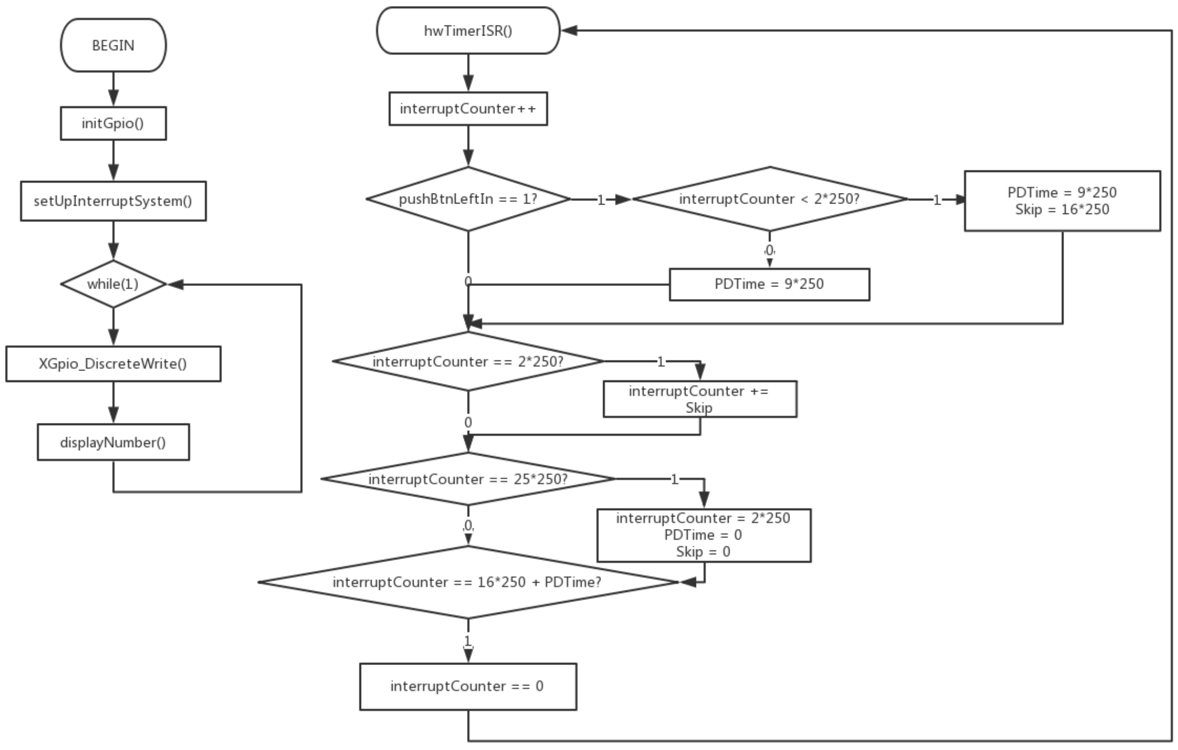
### Board inputs and outputs:

- Left button is used as pedestrian button, which function is explained in detail below.
- VGA output is used to display the traffic light to the monitor and divide the monitor into 12 block regions.
- 7 segment decoder shows how many time left for the current mode, for example, in the every mode of normal period, the clock will count down from 2 to 1. And in the pedestrian light green time, the clock will count down from 5 to 1.
- LEDs display the same mode as monitor display, the first LED light parallel to red light for the traffic light 1, second LED light parallel to yellow light for traffic light 1, third LED light parallel to green light for traffic light 1, 4th LED light parallel to green light for pedestrian light , 5th LED light parallel to red light for traffic light 1, 6th LED light parallel to yellow light for traffic light 2, 7th LED light parallel to green light for traffic light 2. For example, if it is in mode 1R2R, the first and 5th LED light will turn up. The 4th light will only turn up when pedestrian light is green.

**Instruction to the code:**

| Files Name | **Discreption** |
|---|---|
| **Basic.c** | This is the main file of the project. The function of this file is to read the input from FPGA, connect all of the sub files together and output the final result of the whole project. In the main function, the Gpio and interrupt system is setted up and write the outcome of the logic in ISR funtion into 7 VGA display bolcks, LED lights and 7 segment decoder. In the ISR funtion(which is defined in the xinterruptES3.c file), the values needed to outcome are proceed in detailed funtion with respect to input and intern clock.The function of controllers(which are defined in the controller.c) is to calculate the value related to the current system time. In the counter funtion(which is defined in the counter.c), the system time is proceed here related to pedestrain button. |
| **controller.c** | All of the controller logic to control which colour to diplay on the VGA, value on the LEDs and figure on the seven segment decoder. (Separate to Region and SEGLED) |
| **counter.c** | The function that control the clock in the period is defined here |
| **gpio_init.c** | All of the XGpio objects are initialized, including all of the light display on the screen, the LED lights, seven segment decoder and left button |
| **gpio_init.h** | All of the XGpio objects are declared in here |
| **seg7_display.h** | Functions for seven segment decoder display are declared in here |
| **seg7_display.c** | Functions for seven segment decoder display are defined in here. |
| **header.h** | All of the controller logic to control which colour to diplay on the VGA, value on the LEDs, figure on the seven segment decoder and clock processor are declared here. Some colour are also defined here, which will be used in the detialed functions. |

The flow chart is shown below.



In the main function, all the input and output are connected here. In the function, I use interruptCounter as the counter to control the period. Colour_x to represent the variable of the colour show on the monitor, which get value from controller_x() defined in controller_region.c. LED is used to represent the value show on the LED lights, which get value from controller_LED() defined in controller_LEDSEG.c. SEG is used to represent the value show on the seven segment decoder, which get its value from controller_SEG defined in controller_LEDSEG.c. pushBtnLeftIn is a variable used to receive the value from button. If the button is pushed, its value will be one, else its value will be zero. The counter.c gets value from interruptCounter and pushBtnLeftIn as input to calculate how to process the interruptCounter. The logic is as fallow:

The whole period will last for 16 seconds if the pedestrian button is not push. If the pedestrian is pushed in the first two second, the period will prolong to 23 seconds and interruptCounter will jump to 18th second when it reaches the 2nd second, and interruptCounter refresh to the 2nd second when it reaches 23th second. If the pedestrian button is pushed in the other time in the period, the period will prolong to 23 second and interruptCounter will refresh to 2nd second when it reaches 23th second.

| Time(s) | interruptCounter | Light mode (1 2 P) | LEDs |
|---|---|---|---|
| **0-2** | 0-500 | R R R | 1000100 |
| **2-4** | 500-1000 | RY R R | 1100100 |
| **4-6** | 1000-1500 | G R R | 0010100 |
| **6-8** | 1500-2000 | Y R R | 0100100 |
| **8-10** | 2000-2500 | R R R | 1000100 |
| **10-12** | 2500-3000 | R RY R | 1000110 |
| **12-14** | 3000-3500 | R G R | 1000001 |
| **14-16** | 3500-4000 | R Y R | 1000010 |
| **16-18** | 4000-4500 | R R R | 1000100 |
| **18-21** | 4500-5250 | R R G | 1001100 |
| **21-23** | 5250-5750 | R R G(blink) | blink 1001100 |
| **23-25** | 5750-6250 | R R R | 1000100 |

Jump

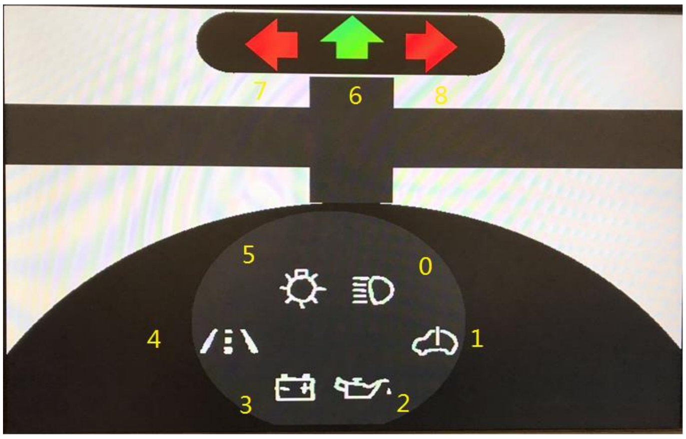
Refresh

## II.Extra

**Introduction and description of the application and its functionalities:**

This is a project of driving game. The situation is that you are driving a motorcycle and encounter a cross road and you can make operation. On the interface, there is a traffic light in the middle top, in which the straight light corresponds to region_6, left light corresponds to region_7, right light corresponds to region_8. On the operation board, there are six signal lights indicating the function of your motorcycle, which correspond to region_0 – region_5. In the middle of the screen, there is a cross road.

The road light operates in that way: RRR -> RYR -> RGR-> RYR -> RRR -> RRY -> RRG -> RRY -> RRR -> YRR ->GRR -> YRR -> RRR -> next period. You should make your option according to traffic light. If you turn up the slide switch 1, the signal light 1 will turn on. So do them for 2,3,4,5,6. The down push button is the speed up button, if you push it, the led lights represents the speed, will increse per second. If you are not push it, the speed will decrease per second. And the seven segment decoder indicate whether you make good choice in the current mode. If the push the button when the matched light is red, then dashes lines "----" will show on the decoder to warn you that you could not do that. If push the button when the matched light is yellow, then dots "...."will show on the decoder to show you that you need to change your decision. If push the button when the matched light is green, then "0000" will show on the decoder to tell you that you make a good decision.



**Board inputs and outputs:**

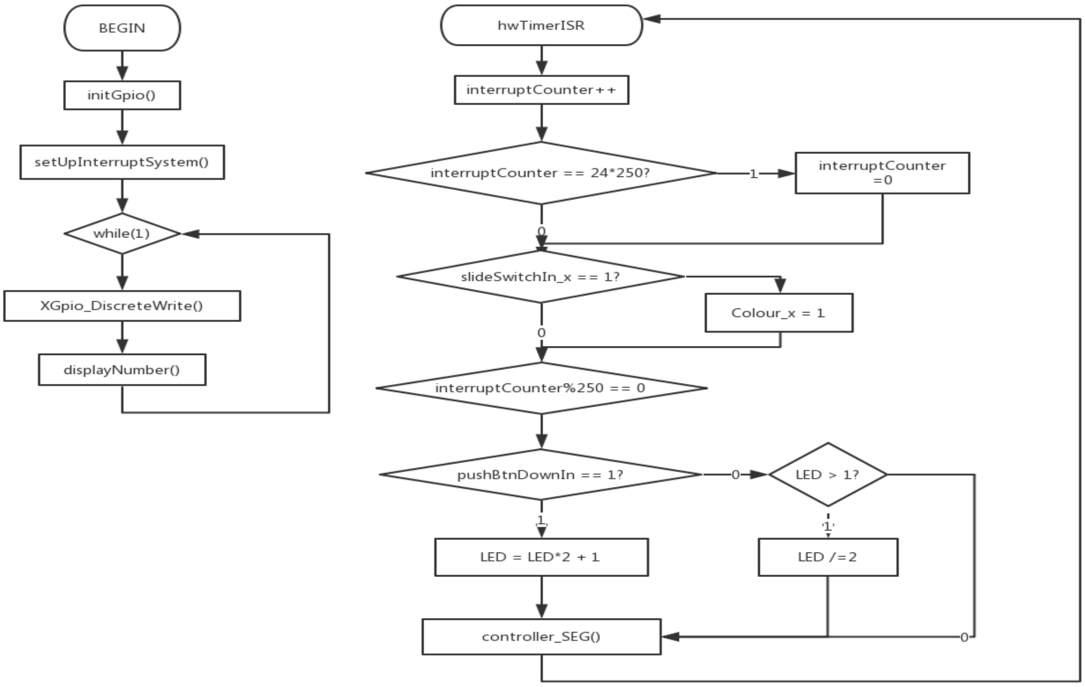
- Left, up, right button refers to turn left, go straight and turn right option.
- VGA output is used to show the traffic light, operation board and cross road.
- Seven segment decoder indicates whether you make a good decision.
- LEDs is used to show the speed of your motor bike.
- Slide Switches used to control the signal lights.

**Instruction to the code:**

| File Name | **Description** |
|---|---|
| **extra.c** | This is the main file of the project. The function of this file is to read the input from FPGA, connect all of the sub files together and output the final result of the whole project. In the main function, the Gpio and interrupt system is setted up and write the outcome of the logic in ISR function into 7 VGA display blocks, LED lights and 7 segment decoder. In the ISR function(which is defined in the xinterruptES3.c file), the values needed to outcome are proceed in detailed funtion with respect to input and intern clock. The function of controllers(which are defined in the controller.c) is to calculate the value related to the current system time. |
| **controller.c** | All of the controller logic to control which color to display on the VGA, value on the LEDs and figure on the seven segment decoder are defined in here. (Separate to Region and SEGLED) |
| **gpio_init.c** | All of the XGpio objects are defined here, including all of the light display on the screen, the LED lights, seven segment decoder, left button, up button, right button, down button and slide switches. |
| **gpio_init.h** | All of the XGpio objects are declared here. |
| **seg7_display.c** | Functions for seven segment decoder display are defined. I almost delete all the original function for seven segment decoder. In my function, the seven segment decoder is used to show whether you make a |

| | |
|---|---|
| | good decision. If the push the button when the matched light is red, then dashes lines "----" will show on the decoder to warn you that you could not do that. If push the button when the matched light is yellow, then dots "...."will show on the decoder to show you that you need to change your decision. If push the button when the matched light is green, then "0000" will show on the decoder to tell you that you make a good decision. |
| **header.h** | All of the controller logic to control which color to display on the VGA, value on the LEDs, figure on the seven segment decoder and clock processor are declared. Some colour are also defined here, which will be used in the detailed functions. |
| **seg7_display.h** | Functions for seven segment decoder display are declared here. And the symbol of dash line, dot and zero are defined here. |

The flow chart is shown below.



In the main function, all the input and output are connected here. In the function, I use interruptCounter as the counter to control the period. Colour_x to represent the variable of the colour show on the monitor, which get value from controller_x() defined in controller_region.c. LED is used to represent the value show on the LED lights, which get value from controller_LED() defined in controller_LEDSEG.c. SEG is used to represent the value show on the seven segment decoder, which get its value from controller_SEG defined in controller_LEDSEG.c. pushBtnLeftIn, pushBtnUpIn, pushBtnDownIn, pushBtnRightIn are variables used to receive the value from buttons. If the button is pushed, its value will be one, else its value will be zero.

The logic for speed up is that: the function will check the value of pushBtnDownIn per second. If the value is one, the LED will times two plus one. If the value is zero, and the LED is larger than one, the LED will divided by two. The logic for traffic light and seven segment decoder is quite simple, just change their values in the period.

That is all about my project. Hope you can enjoy using it ☺.