Traffic Light: User's Manual

Designer: Tianle Zhang s1678924 E4

1. INTRODUCTION

1.1. Basic Function

This project is to display traffic lights for two roads. There are 3 traffic lights for each road: red (R), yellow (Y) and green (G). Each light can be represented by one of the regions in the VGA display. Just as the traffic lights in the UK, these three lights operate as: R → RY → G → Y → R → RY → G → Y →R... (duration is set to 2 s for each light and is displayed on the 7-segment display). The traffic lights for road 1 runs through this cycle first, then those of road 2 runs, and then road 1, and so on. Every time a pedestrian button is pressed, the pedestrian light (PD Go or Stop) will be turned on for 5 seconds indicating that it is time to go across. This time is also displayed on the 7-segment display. Moreover, when the pedestrian has 2 (out of 5) more seconds to go, the pedestrian light will blink 5 times (in these remaining 2 seconds). An indicator LED on the Basys 3 board will light up once the pedestrian button is pressed. This LED will go off once the pedestrian has been allowed to cross. The pedestrian light will be turned on only when both 2 roads are RED.

***Figure1**: Basic display of traffic lights*

2.INPUTS & OUTPUS

2.1. Inputs

Inputs of FPGA board include several switches and 5 buttons (left, right, up, down and centre):

- The left button is the pedestrian button. (In extra features, because I need four buttons to control the direction of the car, the PD button is changed from the left button to the very left switch.)

- The central button is used for resetting.

Some inputs are added in **EXTRA FEATURES**:

- From the left to the right, the first switch now is the pedestrian button.
- The second switch is to choose the mode of the car1, artificial mode or automatic mode.
- The four buttons control the car1 in the artificial mode.

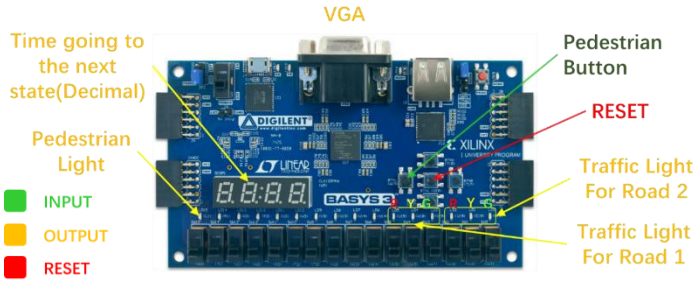

***Figure2**. I/O Ports with Basic Functions*

2.2. Outputs

Outputs of FPGA board include LEDs, 7-segments and VGA:

- The traffic light of road1 is linked with no.3 (green), no.4 (yellow)and no.5 (red) LEDs. The traffic light of road2 is linked with no.0 (green), no.1 (yellow)and no.2 (red) LEDs.
- The pedestrian light is the very left LED.
- 7-segments display the time going to the next state (**Decimal**).
- VGA is used to display traffic lights.

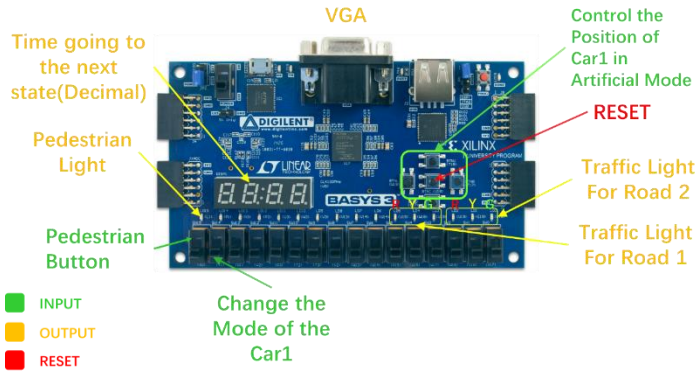

***Figure3**. I/O Ports with Extra Features*

3. FILES, FUNCTIONS AND MACROS

For improving code reusability, modular approach is used. There are few codes in both the main function and the **hwTimerISR ().** All functions are included in different source files, linked by global variables, function calls and variables which are in header files.

Apart from that, macros are used as much as possible to help others to understand.

3.1. Source File and Functions

File Name	Files/Fuctions	Discrption
main.c	int main();	To set up interrupt system, intialize variables, call some packed subfunctions.
control.c	control.c	To control the traffic light. it is a big file to realize the combnition of main function and the interrup system by global varibles.
	void changeMode();	Called by the **hwTimerISR()** to change global varibles to change states of traffic light.
	void setDisplayTime(int period, int order);	To set the display period of a individual state by selecting the time length you want and which state. It is used many times in the **changeMode().**
	void displayLightbyMode(int state);	To display different traffic light states by the input;
	void Button(u16 button);	To avoid bounces, and change pedestrian signal when finishing pressing the left button.
	void transmit(u16 *leddisplay, ...);	To transmit values from control.c to main.c, using pointers to realize, it is an alternative way of defining variables in header files.
interface.c	interface.c	This source file contains convinient functions which can be called directly, without any global variables used.Most of these functions are related to Xgpio
	u16 ledswift(u16 ledvalue, int position, int onoff)	To change the value of a specific bit of the 16-bit integer **ledvalue**
	void turnColour(int column, u16 colour_Up , u16 colour_Mid, u16 colour_Down);	To change the colour at a certain column(1, 2, 3 and 4)
	int getSwitch(int position);	To get the value of a specific bit of the slideswitch.
seg7_display.c	seg7_display.c	A given file related to 7-segments
	displayNumber();	To assign the digit number and the value to be displayed per digit when the timer interrupt occurs.
	calculateDigits();	to extract the digits from the number to be displayed.
	displayDigit();	Called by **hwTimerISR()**, to Select the segments and display the digits on the 7-segment display.
timer_interrupt_func.c	timer_interrupt_func.c	To call the interruption function **hwTimerISR**()
	void hwTimerISR(void *CallbackRef)	Interruption function which excutes codes in it every
xinterruptES3.c	xinterruptES3.c	A given file related to the interruption
platform.c	platform.c	A given file related to the system
gpio_init.c	gpio_init.c	To initialise hardware interfaces.
	XStatus initGpio(void);	To initialise hardware interfaces.
extra.c	extra.c	Contains nearly all extra features used to move the cars.
	void DisplayCarByPosition();	Called in **interrupt system**, changing addresses of the
	void excute(u16 button, XGpio P_BTN, int direction);	To move the car1 according to buttons in the artificial mode
	void Alcar2();	To control the moving of the car2 on the road2
	double speed(double address, double speed_add);	To move cars in a constant speed
	double accelerate (double address, double period, double LeftLimitSpeed, double RightLimitSpeed, double *speed_add);	This function is to move cars in a increasing or decreasing speed with a constant acceleration. acceleration=(RightLimitSpeed-LeftLimitSpeed)/ period
	void ColourCar1();	This function is to colour the car1.

***Table1**. Source File and Functions*

3.2. Header File, Macros and Variables

File Name	Files/Fuctions	Discrption
control.h	**control.h**	It contains nearly all macros and variables across files.
	macros:**RED, YELLOW,GREEN…**	They represent colours for the VGA
	macro: **SECOND**	Represents "* 250", since a second=250 * 0.04 (the time interrupt occurs)
	u16 region_0~ u16 region_8	Represents the colour of each region(0~8)
	double target_x; double target_y; double target2_x; double target2_y;	Represents addresses of two cars.
	int timedisplay	The number displaying on 7-segment
	int mode	Represents the state of the traffic light, it is the input of **displayLightbyMode**();
interface.h	**interface.h**	Contains all declarations of functions in interface.c
gpio_init.h	**gpio_init.h**	Includes all Xgpio variables.
platform.h	**platform.h**	A given file related to the system.
seg7_display.h	**seg7_display.h**	A given file related to 7-segments.
extra.h	**extra.h**	This header file contains nearly all macros and variables across files for extra features.
	macro: **CARWIDTH, CARHEIGHT**	The width and height of cars are 100 and 50 individually
	macro: **UP,DOWN, LEFT, RIGHT**	They represent 1~4, the case number used in choosing directions.
	int carMode	To choose the mode of the car1, artificial mode or the automatic mode

***Table2**. Header File, Macros and Variables*

NB: Those for Extra Features are in dark red, other parts are in black.

4. EXTRA FEATURES

4.1. Hardware

1) A brief road and two traffic lights with round lights can be seen, which is more like a real situation.

   How to realize: change display regions of different colours according to some addresses and simple functions.

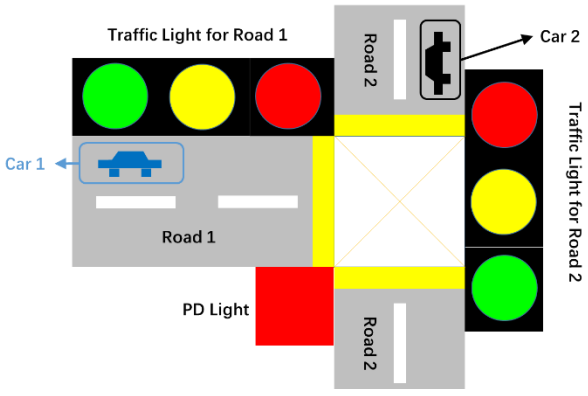

***Figure4**. Extra Features*

2) Two brief cars appear, positions of which are controlled in software.

How to realize: use approaches similar with extra feature 1, but this time, the coordinates of two cars are controlled by spare hardware colour variables.

```
104    assign Y2 = addrh - reg_3_colour;   //let the car2 move horizontally by reg_3_colour and then rotate by 90 degrees
105    assign X2 = addrv - reg_5_colour;   //let the car2 move vertically by reg_5_colour and then rotate by 90 degrees
106    //to be noticed, the order of "Y2" and "X2" is differnt with "X"and"Y". This is done to rotate the car by 90 degrees.
107    assign X = addrh - reg_9_colour;    //let the car1 move horizontally by reg_9_colour
108    assign Y = addrv - reg_10_colour;   //let the car1 move vertically by reg_10_colour
```

***Figure5**. Extra Feature 2: control hardware items by variables assigned in the software*

4.2. Software

1) One of the cars(car2) crosses the road automatically; the other car(car1) can either cross the road by itself, or be controlled by 4 buttons artificially. To be noticed, the two cars will accelerate at the beginning, and decelerate when they want to stop.

How to realize: using the function **accelerate**() in "extra.c" can realize the functionality of speeding up and slowing down with a constant acceleration. In this function, the speed **(a pointer is used)** and the address of the car are changed interrupt by interrupt, so the car seems to move smoothly.

2) Meanwhile, the car2 is black, and the colour of car1 depends on whether it obeys the traffic rule or not. When the traffic light for road 1 is not green and the car2 is on the cross, the colour of it will be red, otherwise the colour will be blue. Therefore, the car1, in the automatic mode, is always blue.

How to realize: change the colour by the position of the car and the colour of the traffic light.

3) The 7-segments can display decimal numbers.

How to realize: some "if-else" conditions about leading zeros, decimal point and maximum displayed numbers are added in "seg7_display.c". Meanwhile, the number put into **displayNumber()** is multiplied by 10.

4) The period of each individual state of traffic lights is changed. Red lights, yellow lights and green lights last for 2s, 1s and 5s separately. (The time can be changed easily by altering parameters of function **setDisplayTime()** in "control.c".

How to realize: a global integer array **timereg**[20] is defined to store periods of different states of traffic lights(these periods are set by inputs of the function). By this array, some calculation of both up limits and down limits of display time are done.

5.REFERANCE:

- Arslan, T. (2016). *Engineering Software 3: Assessment 2 Traffic Light.* Retrieved on 25th November, 2016 from https://www.learn.ed.ac.uk/webapps/portal/execute/tabs/tabAction?tab_tab_group_id=_1_1