Traffic Lights User and Programmer's Guide

- *User Guide*

This project implemented three roads with traffic lights (three lights: red, yellow and green for each road) and one pedestrian light. The traffic lights on each road will looping from red as shown in Fig.1. Every state of lights will remain for 1 second and then move to the next state. Same as reality, only one road can run the loop at the same time, the loop will begin from the first road after power up, that is, the whole loop of the three road will be: *All Three Roads Red -> 1st Road Red+ Yellow, 2nd & 3rd Road Red -> 1st Road Green, 2nd & 3rd Road Red -> 1st Road Yellow, 2nd & 3rd Road Red -> All Three Roads Red -> 2nd Road Red+ Yellow, 1st & 3rd Road Red- > 2nd Road Green, 1st & 3rd Road Red -> ...*

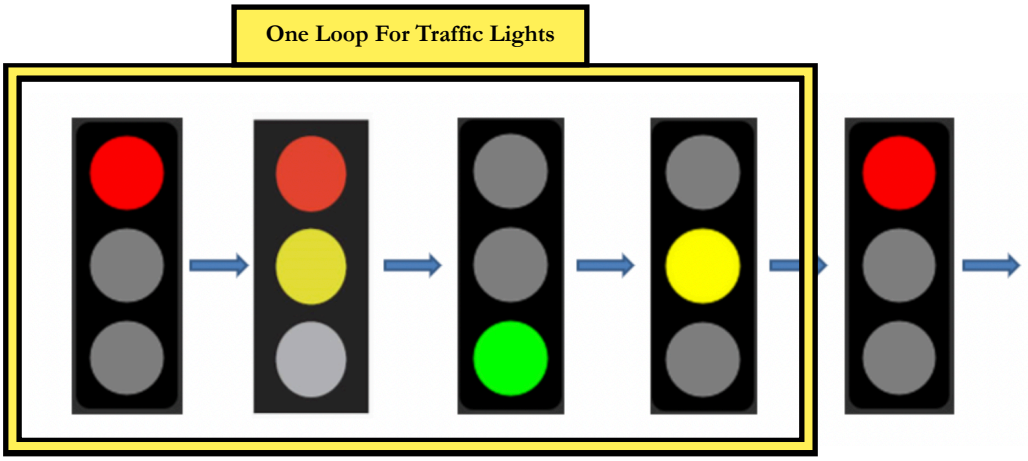

Fig.1 Traffic Lights' Loop (Red -> Red+Yellow -> Yellow -> Green -> Yellow -> Next Loop...)

When the traffic lights looping, The **state of lights** on all three roads will be displayed on **LEDs** on the FPGA board (Fig.2) as well as the **monitor** (Fig.3) connected with the FPGA board via VGA cable. The **duration of current light** and **current road number** will be display on the four-digit-display.

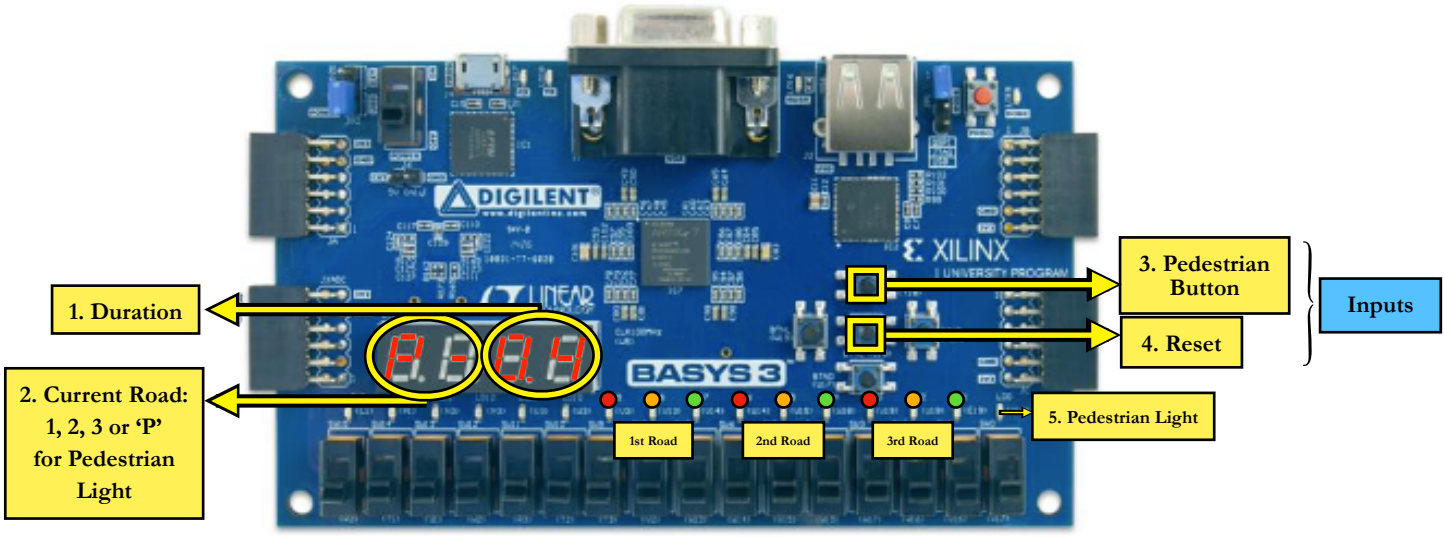

Fig.2 Basys 3 FPGA Board with Comments for Traffic Lights Display

Once the **3. Pedestrian Button** be pressed, the **5. Pedestrian Button** turns on until the loop at current road finished, then all three roads will display red and the pedestrian light turns green for 5s and will blink 5 times at the last 2 seconds. The **4-digit-display** will display the duration **from 5 to 0** and display **'P'** which indicates the pedestrians are currently allowed to cross. After pedestrian light turns red, the three roads will continue the states before the pedestrian light turns green.

- *Extra features*

There are 4 extra features implemented in this project:

1. *Pedestrian Priority:* If this slide switch is on, once the pedestrian button is pressed, the program will allow pedestrians to cross **immediately** and then go back to the last state after the pedestrian light turns off. This feature acts as the Yellow Flashing light in UK.
2. *Pause:* If this slide switch is on, the program will be **pause** until it switch off. It can be used, for example, if 1st road is super busy while others are not, you can use this feature when 1st road is green, in order to allow more vehicle across.
3. *Stop All: If* this slide switch is on, all lights will t**urn red immediately** until it switch off. It can be used in emergency.
4. *Change traffic and pedestrian lights' durations:* To use this function, you should first switch on the left most slide switch and then using

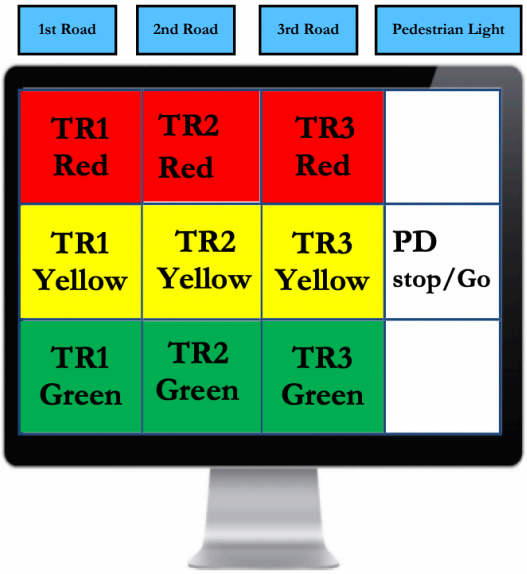

Fig.3 VGA Display Regions for Traffic Lights

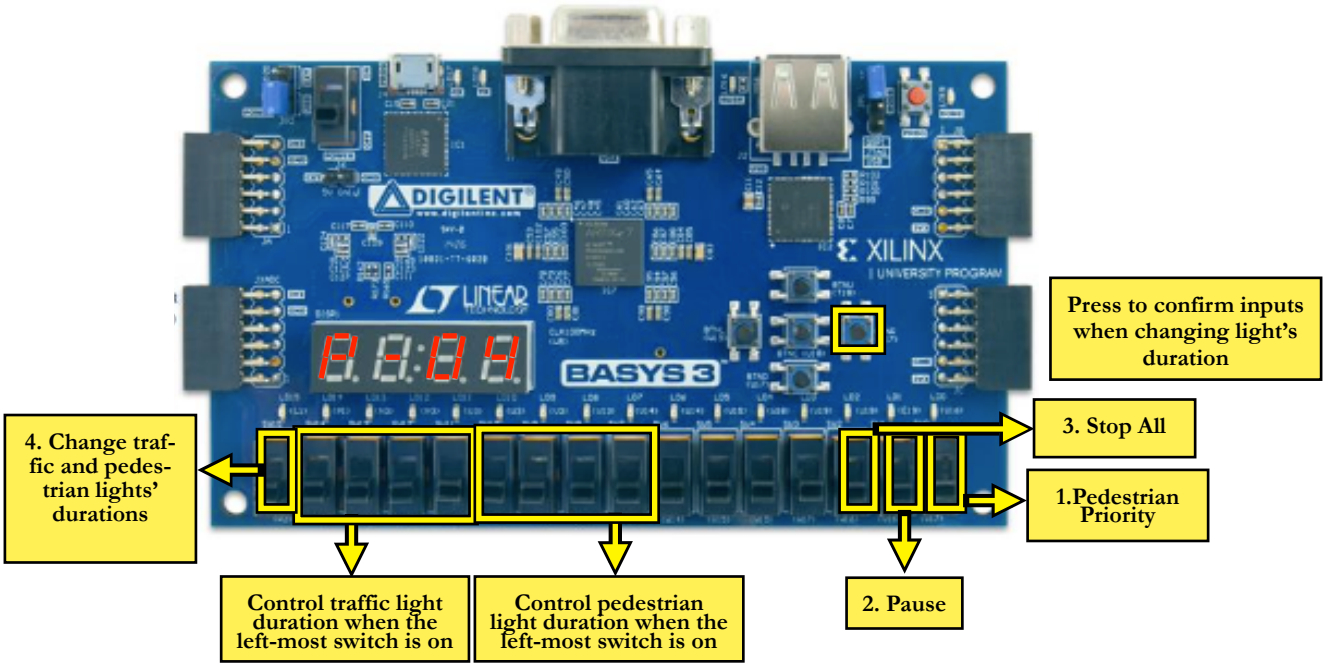

Fig.4 Basys 3 FPGA Board for extra features.

LED 12-15 to input the new light duration you want, and using LED 8-11 to input the new pedestrian light duration you want (As stated in Fig.4) , finally press the right button BTNR to confirm. The new duration will be applied in the next loop.

Note: If either of inputs is zero, the program will consider the inputs are invalid and will not change the duration. One usage of this feature could be: you can set a longer duration at busy time and a shorter duration at night to decrease waiting time.

- Programmer's guide

Main process(main.c)

The main process of the program is shown in Fig.5. After the traffic lights initialised, the program will display lights' loop (Fig.1) on one road, check whether the pedestrian button is pressed and then move to the next road or display pedestrian light depending on the state of pedestrian button.

Display traffic lights for the road contains: 1. VGA display for traffic lights. 2. LED display for traffic light state. 3. 4-digit-7-segment display for duration and road number. 4. Keep checking pedestrian button, turn on pedestrian LED if the button is pressed.

Display pedestrian light contains: 1. VGA display for traffic lights and pedestrian light including the blinking at the last 2 seconds. 2. Turn off pedestrian LED on the board. 3. 4-digit-7-segment display for duration and character 'P'.

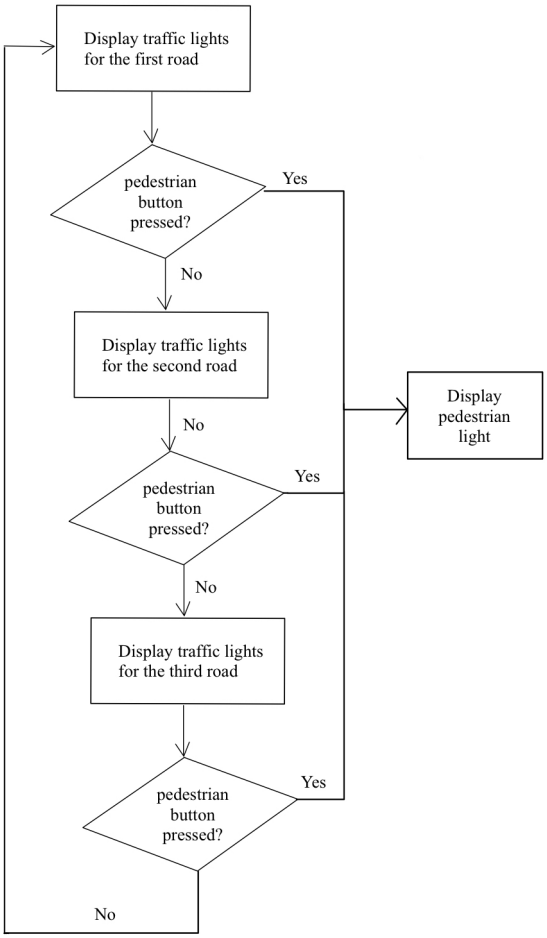

Fig.5 Flowchart of Program

Modules

Overview

There are 9 modules in this project, *seg7_display, vga_display* and *led_display* for displaying; *traffic_lights* contains the definition of traffic light structures and functions used to switch lights; *display_control* contain functions used in the main process (as shown in Fig.5) of this project; *timer_interrpt_func* contains ISR used to interact with interrupt system; *platform* and *xinterruptES3* are provided in lab. The relationship between modules are shown in Fig.6.

traffic_lights

To increase the readability of codes, three new structures have been defined in this project. The summary of the structures are shown below (Table.1) . Those structures will be used in functions in this module and the display_control module. The functions in this module is summarised in table 2. The first and third function are responsible for initialise and the second function is used to read the information store in the road structure and then display current lights states to monitor.

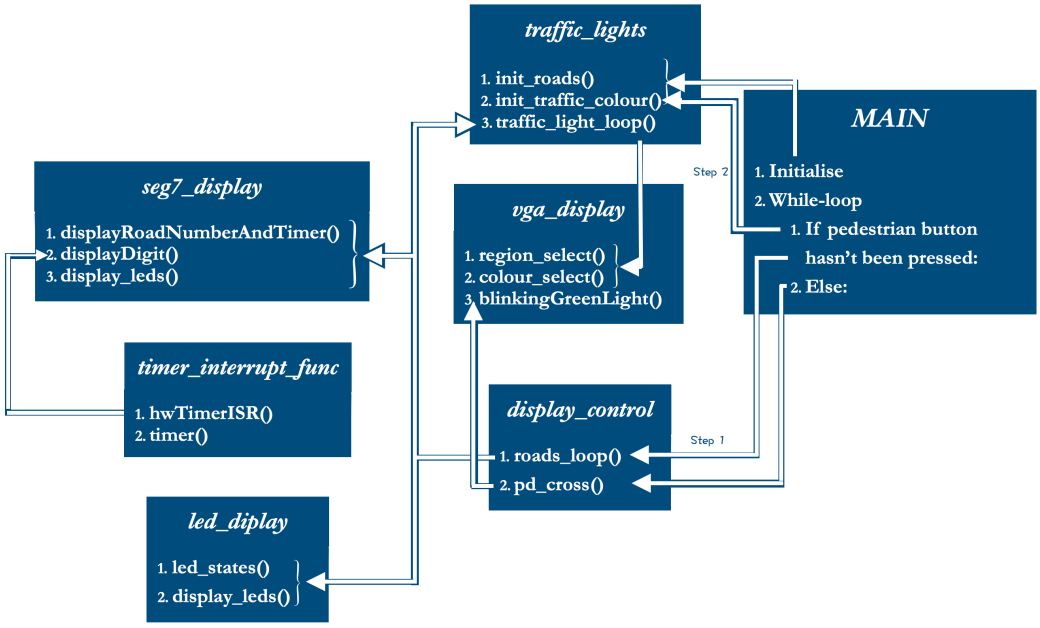

Fig.6 Modules of Program

NAME	STRUCTURE	DESCRIPTION
struct trafficStates	char red_light_state; char yellow_light_state; char green_light_state;	> States of three traffic lights. 'r' for red, 'w' for white, y' for yellow, 'g' for green.
struct road	u8 red_light_address; u8 yellow_light_address; u8 green_light_address; u16 red_light_led_address; u16 yellow_light_led_address; u16 green_light_led_address; u8 curr_state_index; struct trafficStates *states_sequence;	> Location of three traffic lights on VGA and LEDs, address in VGA is a number between 0 to 11 corresponding to 12 regions shown in Fig., LEDs' address is a 16 bits binary number corresponding to 16 LEDs on board. > Current index of state in the state array > The state array, like {R, RY, G, Y, R}
struct pedestrianLight	u8 light_vga_address; u16 light_led_address;	> Locations of pedestrian light on VGA and LEDs

Table.1 New defined structures

NAME	SYNTAX	DESCRIPTION
init_traffic_colour()	void init_traffic_colour (struct road* rd, s8 numOfRoads, struct pedestrianLight* ped_light)	Initialise traffic lights and pedestrian light (all reds).
traffic_light_loop()	void traffic_light_loop (struct road rd)	Display current state of traffic lights (store in states_sequence) to monitor.
init_roads()	void init_roads(struct road* roads, struct trafficStates* lights_operation, struct pedestrianLight* ped_light)	Initialise three roads and pedestrian light with: regions on VGA & LEDs display, and state sequence.

Table.2 Functions in traffic_lights

seg7_display

This module contains functions used to display digit on 4-digit-7-segment-display, which convert the three inputs: roadNum, number and pointBool to 7-segment code and select location to display.

NAME	SYNTAX	DESCRIPTION
displayRoadNumberAndTimer()	void displayRoadNumberAndTimer(u8 roadNum, u16 number, u8 PointBool)	Based on previous lab code, this function take 3 inputs and will update the digit array to 1. A number of road/ 'P' for pedestrian light, 2. A dash sign 3. A 2 bits digit for duration.
displayDigit()	void displayDigit()	Added character 'P' to case-switch, display decimal point at 3rd digit if PointBool is TRUE.

Table.3 Functions in seg7_display

vga_display

This module contains two select functions allowing user to use integer 0-11 and character like 'r' to select colour and region of VGA display.

NAME	SYNTAX	DESCRIPTION
region_select()	XGpio region_select(u8 region_selector)	Return the XGpio object of VGA region corresponding to the integer (0-11) inputed.
colour_select()	u16 colour_select(char colour_char)	Return 12 bits colour code corresponding to the char inputed.
blinkingGreenLight()	void blinkingGreenLight(char* colour)	Change char to 'g' standing for colour green if input is 'w' and vice versa.

Table.4 Functions in vga_display

The 3rd function is a helper function for blinking of the pedestrian light.

led_display

This module contains a main function display_leds() which take roads and pedestrian light structure as inputs and output the corresponding LEDs. The function led_states() is a helper function used to get 16bits led states which will do logic OR to all LED states stored in structure road and pedestrian light.

NAME	SYNTAX	DESCRIPTION
led_states()	u16 led_states(struct road rd)	Return the led states (16bits) from road.
display_leds()	void display_leds(u8 ped_btn, struct road* roads, u8 numOf-Roads,struct pedestrianLight* ped_light);	Display the LEDs state of all roads.

Table.5 Functions in led_display

display_control

This module contains the two main functions(Table.6) in this project. The first one is used to complete one traffic light loop(Fig.1) in one single road, which contains VGA, Seg7 and LEDs display, and will keep updating the state of pedestrian button. The second function is the function used to active VGA, Seg7 and LEDs displays when pedestrian are allowed to cross.

NAME	SYNTAX	DESCRIPTION
roads_loop()	void roads_loop(struct road* roads, u8 rd_num, struct pedestrianLight* ped_light)	Main loop for this project which contains VGA, Seg7 and LEDs display for traffic lights.
pd_cross()	void pd_cross(struct pedestrianLight* ped_light)	Main loop for pedestrian light which contains VGA, Seg7 and LEDs display for pedestrian light.

Table.6 Functions in display_control

There are 4 variables in this module, which declared in header file. The first two variables allow user to change the number of roads and traffic light states in the future. The 3rd one used to transmit the states of pedestrian button among different modules to allow other actions. The last variable allow user to change the duration of pedestrian light in the future.

NAME	DESCRIPTION
u8 len_of_states_array	Integer for the length of states sequence, for example, in this coursework it should be 4 for the state sequence{R, RY, Y, G}.
u8 num_of_roads	Integer for the number of roads, it should be 3 in this coursework.
u8 ped_btn	Boolean for whether the pedestrian button has been pressed
u8 light_counter	Duration for traffic lights, could be modified to control the duration of traffic lights.
u8 pd_counter	Duration for pedestrian light, could be modified to control the duration of pedestrian light.

Table.7 Variables in display_control

timer_interrpt_func

NAME	SYNTAX	DESCRIPTION
hwTimerISR()	void hwTimerISR(void *CallbackRef)	Interrupt Service Routine which serving interrupts.
timer()	void timer()	Count real life time using counter.

Table.8 Functions in timer_interrupt_func

The second function contain a counter which will be increased at each call, and a boolean *timer_served* which will be set to TRUE when the counter equal to REQUIRED_TIME/INTERRUPT_SYSTEM_TIME, in this project REQUIRED_TIME = 0.01 for 1 millisecond and INTERRUPT_SYSTEM_TIME = 0.004 as set in hardware.

This module calls timer() and displayDigit() (in seg7_display module) in the hwTimerISR() function to increase counter

NAME	DESCRIPTION
u8 timer_served	Boolean for the timer (served or not), will be modified in timer().
u8 interruptCounter	System counter, which used to count time and will be increased at each interrupt (in hwTimerISR()).

Table.9 Variables in timer_interrpt_func

and display digit at each interrupt.

There are two variables defined in the header file of this module, they are all used to transmit the states of time counter among different modules to allow other actions.

Extra features

All extra functions are added in display_control .c. The summarize of functions are shown below. The last function used pointer to change the variable in main.c directly in order to change the duration. The program call all the four functions in both

NAME	SYNTAX	DESCRIPTION
checkYellowFlashing()	void checkYellowFlashing(u8* light_var_ct,u8* pd_var_ct,struct road* roads,struct pedestrianLight* ped_light,u8 ped_btn)	Keep checking the state of the slide switch in, if it is 1 and ped_btn is on, then allow pedestrians to cross immediately
checkPause()	void checkPause(u8 rd_num,s8 ms)	Keep checking the state of the slide switch in, while it is 2, pause the program and keep displaying numbers
checkAllRed()	void checkAllRed(struct road* roads,struct pedestrianLight* ped_light)	Keep checking the state of the slide switch in, while it is 4, turn all lights to red and display dash.
checkChangeNextSpeed()	void checkChangeNextSpeed(u8* light_var_ct,u8* pd_var_ct)	Keep checking the state of the slide switch in, if the left most switch is on, take slides 12-15 as input to traffic light duration and slides 8-11 as input to pedestrian light duration.

Table.10 Extra feature functions in display_control.c

roads_loop() and pd_cross() to keep checking when should the program perform extra feature.