**Lab 1 Blink LED – ECE 5780**

Nate Sheffield

A02268057

Nathan Critchfield

A02283426

**Objective**

The purpose of this lab is to implement a toggling LED on the STM32 Nucleo-L476RG Board when a button is pressed using FreeRTOS.

**Procedure**

This lab started by installing the Keil uVision development environment. We installed the environment and the packages needed for development on the STM32 board. Once we had the environment setup, we looked into using FreeRTOS to complete the assignment. With FreeRTOS setup we made two threads. One controlled the LED and one read a push button input. The threads were connected through a global variable. Once all of that was setup correctly, we were able to push the button to toggle the LED on and off.

**Results**

Since this is the first lab of the semester we had to go through several layers of setup in order to produce the results we wanted for the lab. We first had to download and setup keil μVision 5 which is no simple task. We had a few minor issues getting the environment setup but after a short while we were able to get everything setup to start coding. After this we were finally able to code our simple main file to create two tasks to run the button and LED and then monitor a global variable to toggle the led state to turn it on and off. The coding part of this was fairly straight forward as this is a simple result to produce for a lab. However, we had several issues with getting the right configuration files connected, setup and modified to build our code and then run that code on the board. We had an issue that our code would not process our button input because we were not aware that we needed to modify the configuration file to use heap4 instead of heap3. After changing this value our code worked as intended and the LED was able to toggle on and off with button presses.

A white electronic board with black wires and a red light

Description automatically generatedA white electronic board with a red light

Description automatically generated

**Figure 1.** STM32 Board with LED toggled on and off after button presses

**Conclusion**

In conclusion, we were able to setup Keil μVision 5 to build and run our c code on our STM32 Nucleo board using FreeRTOS. After creating two different tasks to run the button and LED we then made two different functions to modify a global variable to toggle the state of the LED on and off with button presses. After debugging our issues with the code and in the setup of the Keil environment we were able to turn the LED on and off with button presses on the board to complete the lab requirements.

**Appendix**

***Blink\_LED code***

1. #include "FreeRTOS.h"

2. #include "stm32l476xx.h"

3. #include "system\_stm32l4xx.h"

4. #include "task.h"

5. #include "timers.h"

6. #include "stdint.h"

7.

8. static uint32\_t led\_state;

9. void LED(void \*pvParameters);

10. void Button(void \*pvParameters);

11.

12. //Function to toggle led\_state

13. void LED(void \*pvParameters){

14. while(1){

15. //If the LED is on turn it off

16. if(led\_state == 1){

17. GPIOA->BSRR |= GPIO\_BSRR\_BS5;

18. }

19. //If the LED is off turn it on

20. else {

21. GPIOA->BSRR |= GPIO\_BSRR\_BR5;

22. }

23. }

24. }

25.

26. //Function to read in button state and led\_state

27. void Button(void \*pvParameters){

28. while(1){

29. uint32\_t button\_in;

30. //Read in the value of the button

31. button\_in = GPIOC->IDR;

32. button\_in &= GPIO\_IDR\_ID13\_Msk;

33.

34. //If the button is pressed toggle the LED

35. if(button\_in == 0){

36. while(button\_in == 0){

37. button\_in = GPIOC->IDR;

38. button\_in &= GPIO\_IDR\_ID13\_Msk;

39. }

40. if(led\_state == 0){

41. led\_state = 1;

42. }

43. else {

44. led\_state = 0;

45. }

46. }

47. }

48. }

49.

50. int main(void) {

51. //Initialize System

52. SystemInit();

53.

54. //Turn Clock on GPIOA and GPIOC

55. RCC -> AHB2ENR |= RCC\_AHB2ENR\_GPIOAEN;

56. RCC -> AHB2ENR |= RCC\_AHB2ENR\_GPIOCEN;

57.

58. //Set PA5 to output mode for LED

59. GPIOA->MODER &= ~GPIO\_MODER\_MODE5\_1;

60. GPIOA->MODER |= GPIO\_MODER\_MODE5\_0;

61. //Turn LED on

62. GPIOA->BSRR |= GPIO\_BSRR\_BS5;

63. led\_state = 1;

64.

65. //Set PC13 to input mode for Button

66. GPIOC->MODER &= ~GPIO\_MODER\_MODE13; //0xf3ffffff

67.

68.

69.

70. xTaskCreate( //Task for LED

71. LED,

72. "LED",

73. 16,

74. NULL,

75. 1,

76. NULL);

77.

78. xTaskCreate( //Task for Button

79. Button,

80. "Button",

81. 16,

82. NULL,

83. 1,

84. NULL);

85.

86. //Start Task Scheduler

87. vTaskStartScheduler();

88. while(1);

89. }

90.