Department of Engineering Technology and Industrial Distribution

ESET 349 Microcontroller Architecture

Lab 3: Toggling LEDs using assembly language programming

Objectives

1. Build a simple electric circuit on a breadboard with LEDs and resistors in series.
2. Develop a flowchart for programming and use this flowchart to implement the flowchart in assembly language.
3. Use function calls and the link register to decide the flow of control.
4. Become familiar with GPIO memory mappings and utilize the STM32F401RE Specifications document to identify addresses and offsets for programing ports.

Your Tasks

1. Program an STM32F401RE microcontroller to toggle three LEDs at a given interval. The program configures three pins as output pins. Use PA5, PA6, and PA7 as output pins in your program if they are functional. These pins are connected to a circuit placed on a breadboard. Each LED is connected in series with a 330Ω/220 Ω resistor. Figure 1 shows a sketch. The program repeatedly sends HIGH and LOW signals at a predetermined time interval such that the LEDs toggle in a **sequence**.

2. Implement a two-loop delay routine as described in the flowchart on the next page.

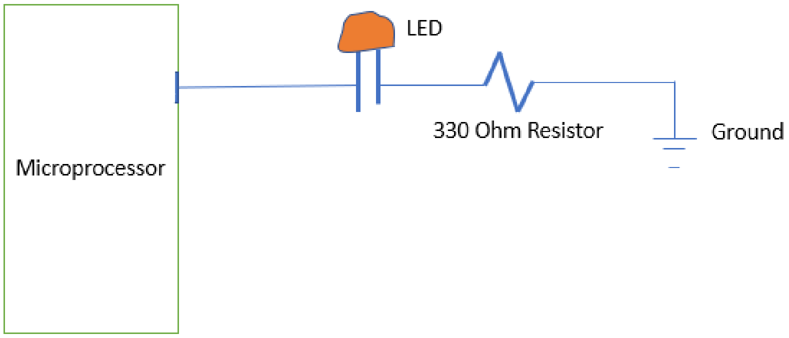


Figure 1. Circuit sketch

Flowchart

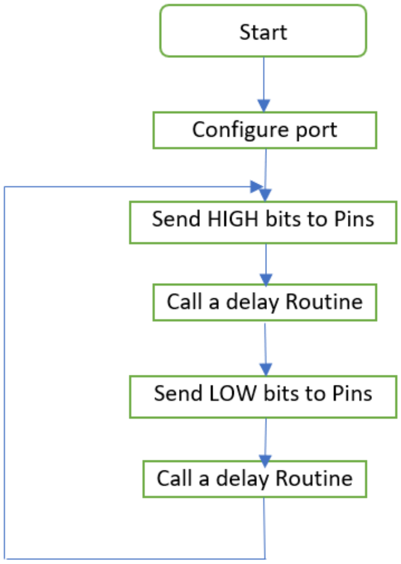


Figure 2. A flowchart for the main body of the program

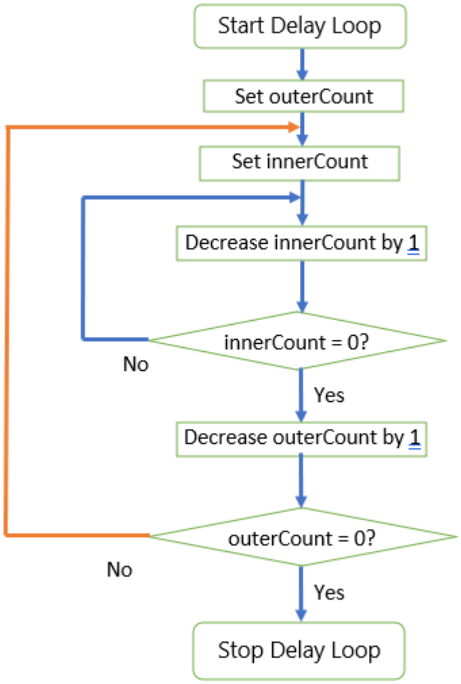


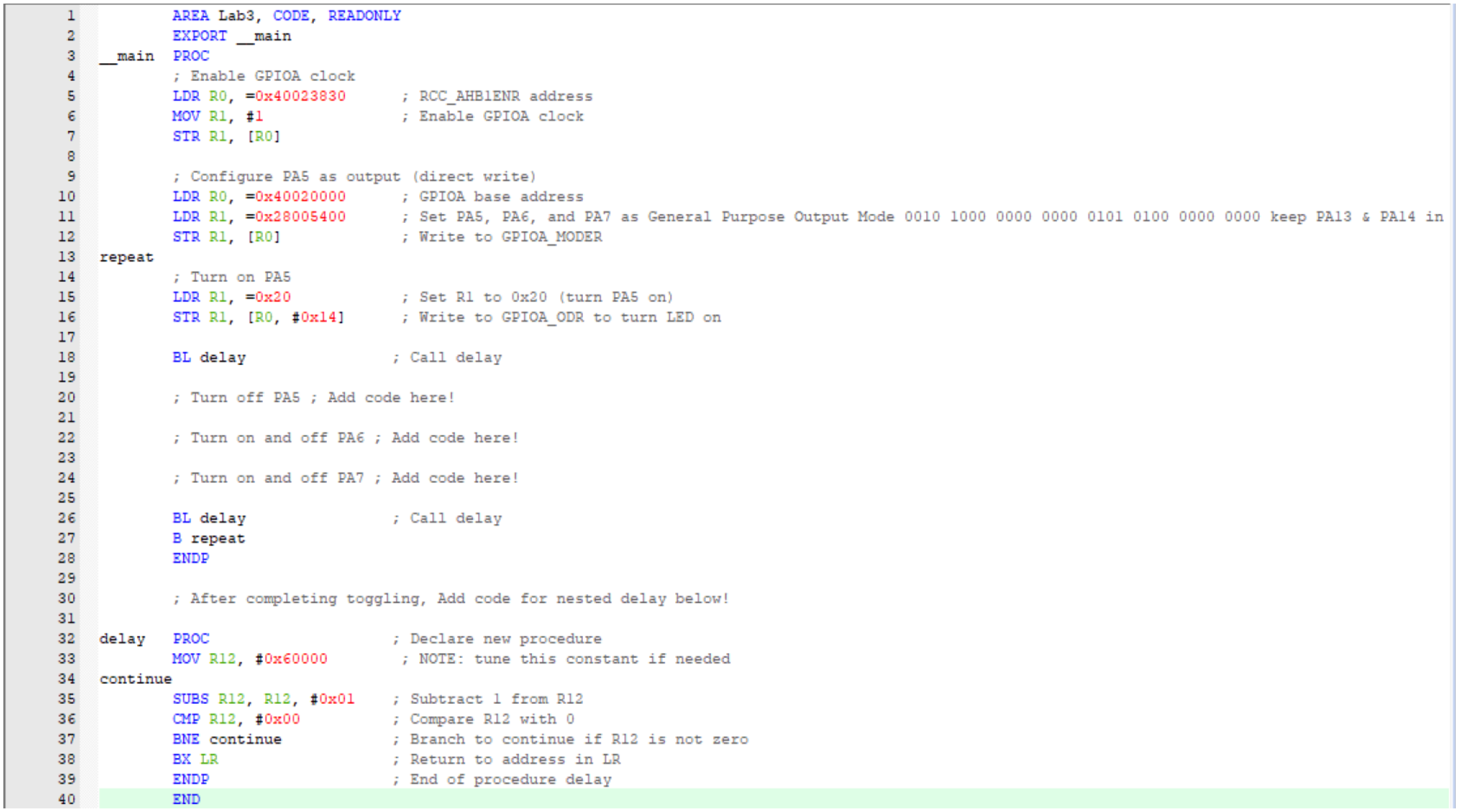
Figure 3. A flowchart for the two-loop (nested) delay routine

Program Sketch

Incomplete program as a guide only; please complete the code. Notice this skeleton program sends a HIGH signal to PA5 only. If you suspect your hardware is not functional, try using a different set of three pins, or reach out to your TA for assistance.

We encourage you to come up with the entire program and use this snippet only as a reference.

Complete toggling with the given single-loop delay function. After demonstrating to your TA, replace it with the two-loop delay function as suggested in Figure 3.



```
        AREA Lab3, CODE, READONLY
        EXPORT __main
__main  PROC
        ; Enable GPIOA clock
        LDR R0, =0x40023830      ; RCC_AHB1ENR address
        MOV R1, #1               ; Enable GPIOA clock
        STR R1, [R0]

        ; Configure PA5 as output (direct write)
        LDR R0, =0x40020000      ; GPIOA base address
        LDR R1, =0x28005400      ; Set PA5, PA6, and PA7 as General Purpose Output Mode 0010 1000 0000 0000 0101 0100 0000 0000 keep PA13 & PA14 in
        STR R1, [R0]             ; Write to GPIOA_MODER
repeat
        ; Turn on PA5
        LDR R1, =0x20            ; Set R1 to 0x20 (turn PA5 on)
        STR R1, [R0, #0x14]      ; Write to GPIOA_ODR to turn LED on

        BL delay                ; Call delay

        ; Turn off PA5 ; Add code here!

        ; Turn on and off PA6 ; Add code here!

        ; Turn on and off PA7 ; Add code here!

        BL delay                ; Call delay
        B repeat
        ENDP

        ; After completing toggling, Add code for nested delay below!

delay   PROC                    ; Declare new procedure
        MOV R12, #0x60000        ; NOTE: tune this constant if needed
continue
        SUBS R12, R12, #0x01    ; Subtract 1 from R12
        CMP R12, #0x00          ; Compare R12 with 0
        BNE continue            ; Branch to continue if R12 is not zero
        BX LR                   ; Return to address in LR
        ENDP                    ; End of procedure delay
        END
```