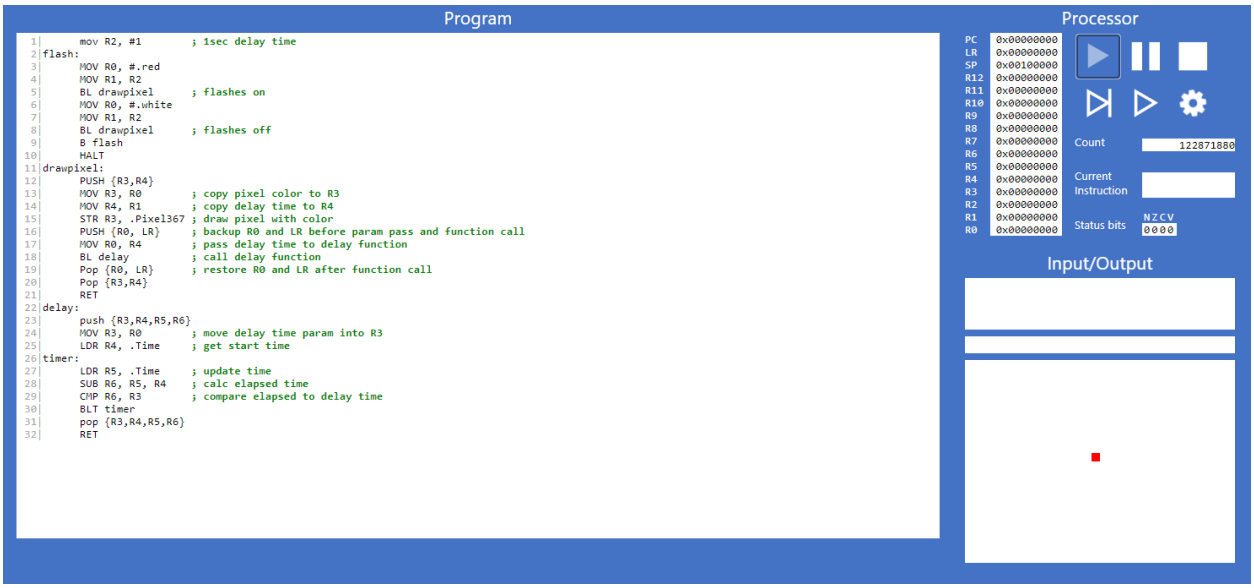
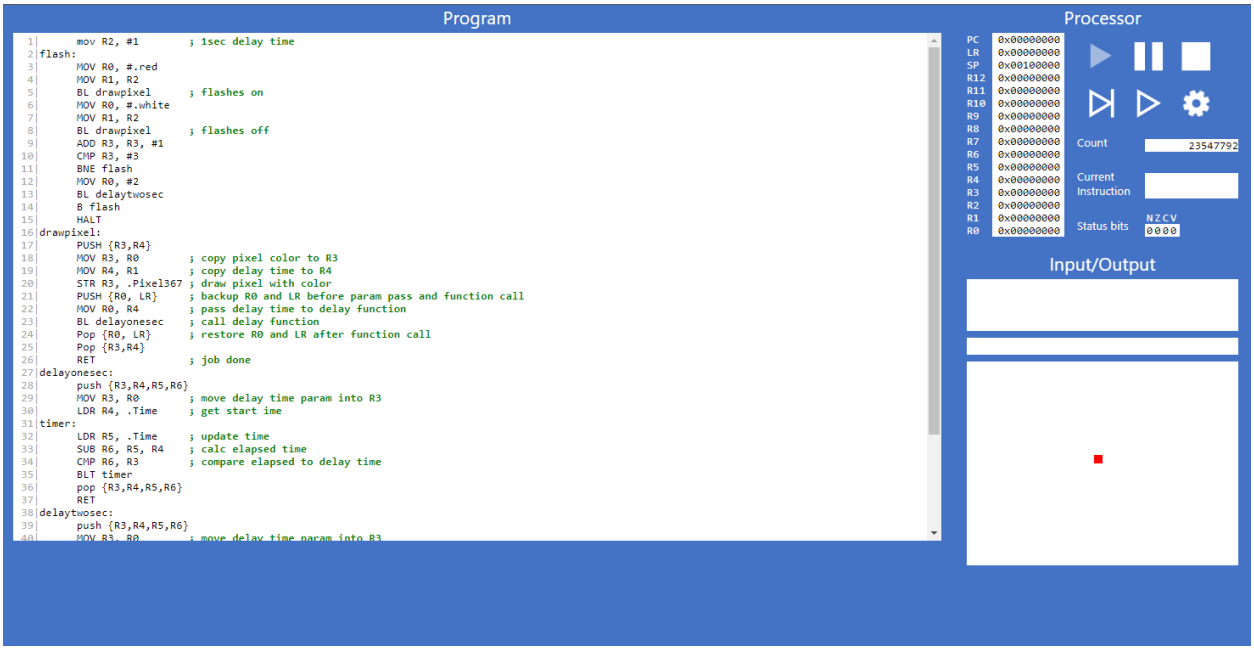
Lab 10 – Computer Systems (Mohamed Shafi Uzman Fassy - 102608927)

10.1

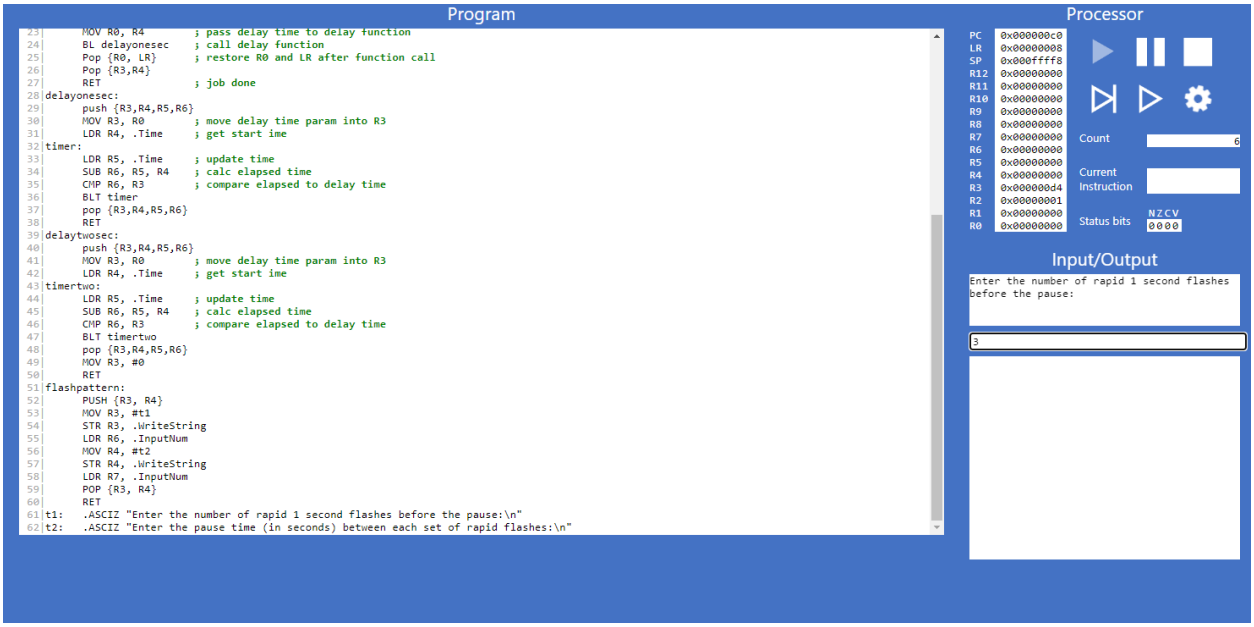


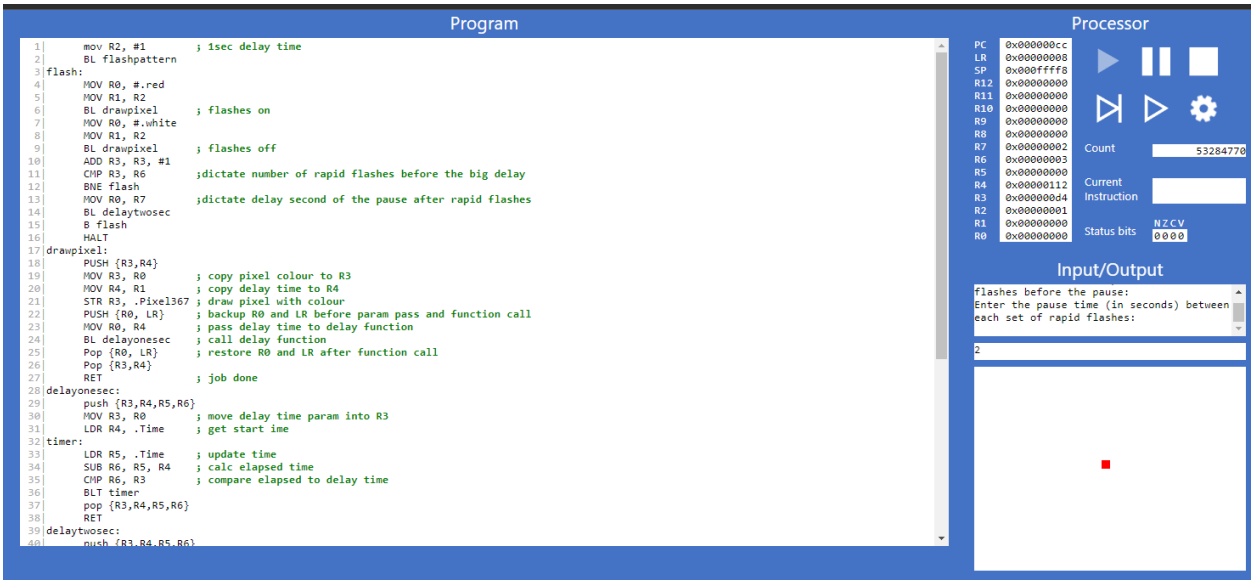
c) we use LR to push it into the stack so that after the nesting function is completed it will know the address to return back to and execute the next instruction.

10.2



10.3


Program
MOV R0, R4 ; pass delay time to delay function
BL delayonesec ; call delay function
Pop {R0, LR} ; restore R0 and LR after function call
Pop {R3,R4}
RET ; job done
delayonesec:
push {R3,R4,R5,R6}
MOV R3, R0 ; move delay time param into R3
LDR R4, .Time ; get start ime
timer:
LDR R5, .Time ; update time
SUB R6, R5, R4 ; calc elapsed time
CMP R6, R3 ; compare elapsed to delay time
BLT timer
pop {R3,R4,R5,R6}
RET
delaytwosec:
push {R3,R4,R5,R6}
MOV R3, R0 ; move delay time param into R3
LDR R4, .Time ; get start ime
timertwo:
LDR R5, .Time ; update time
SUB R6, R5, R4 ; calc elapsed time
CMP R6, R3 ; compare elapsed to delay time
BLT timertwo
pop {R3,R4,R5,R6}
MOV R3, #0
RET
flashpattern:
PUSH {R3, R4}
MOV R3, #t1
STR R3, .WriteString
LDR R6, .InputNum
MOV R4, #t2
STR R4, .WriteString
LDR R7, .InputNum
POP {R3, R4}
RET
t1: .ASCIZ "Enter the number of rapid 1 second flashes before the pause:\n"
t2: .ASCIZ "Enter the pause time (in seconds) between each set of rapid flashes:\n"
Processor
PC 0x000000c0
LR 0x00000008
SP 0x000ffff8
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x000000d4
R2 0x00000001
R1 0x00000000
R0 0x00000000
Count 6
Current Instruction
Status bits NZCV 0000
Input/Output
Enter the number of rapid 1 second flashes before the pause:
3


Program
mov R2, #1 ; 1sec delay time
BL flashpattern
flash:
MOV R0, #.red
MOV R1, R2
BL drawpixel ; flashes on
MOV R0, #.white
MOV R1, R2
BL drawpixel ; flashes off
ADD R3, R3, #1
CMP R3, R6 ;dictate number of rapid flashes before the big delay
BNE flash
MOV R0, R7 ;dictate delay second of the pause after rapid flashes
BL delaytwosec
B flash
HALT
drawpixel:
PUSH {R3,R4}
MOV R3, R0 ; copy pixel colour to R3
MOV R4, R1 ; copy delay time to R4
STR R3, .Pixel367 ; draw pixel with colour
PUSH {R0, LR} ; backup R0 and LR before param pass and function call
MOV R0, R4 ; pass delay time to delay function
BL delayonesec ; call delay function
Pop {R0, LR} ; restore R0 and LR after function call
Pop {R3,R4}
RET ; job done
delayonesec:
push {R3,R4,R5,R6}
MOV R3, R0 ; move delay time param into R3
LDR R4, .Time ; get start ime
timer:
LDR R5, .Time ; update time
SUB R6, R5, R4 ; calc elapsed time
CMP R6, R3 ; compare elapsed to delay time
BLT timer
pop {R3,R4,R5,R6}
RET
delaytwosec:
Processor
PC 0x000000cc
LR 0x00000008
SP 0x000ffff8
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000002
R6 0x00000003
R5 0x00000000
R4 0x00000112
R3 0x000000d4
R2 0x00000001
R1 0x00000000
R0 0x00000000
Count 53284770
Current Instruction
Status bits NZCV 0000
Input/Output
flashes before the pause:
Enter the pause time (in seconds) between each set of rapid flashes:
2