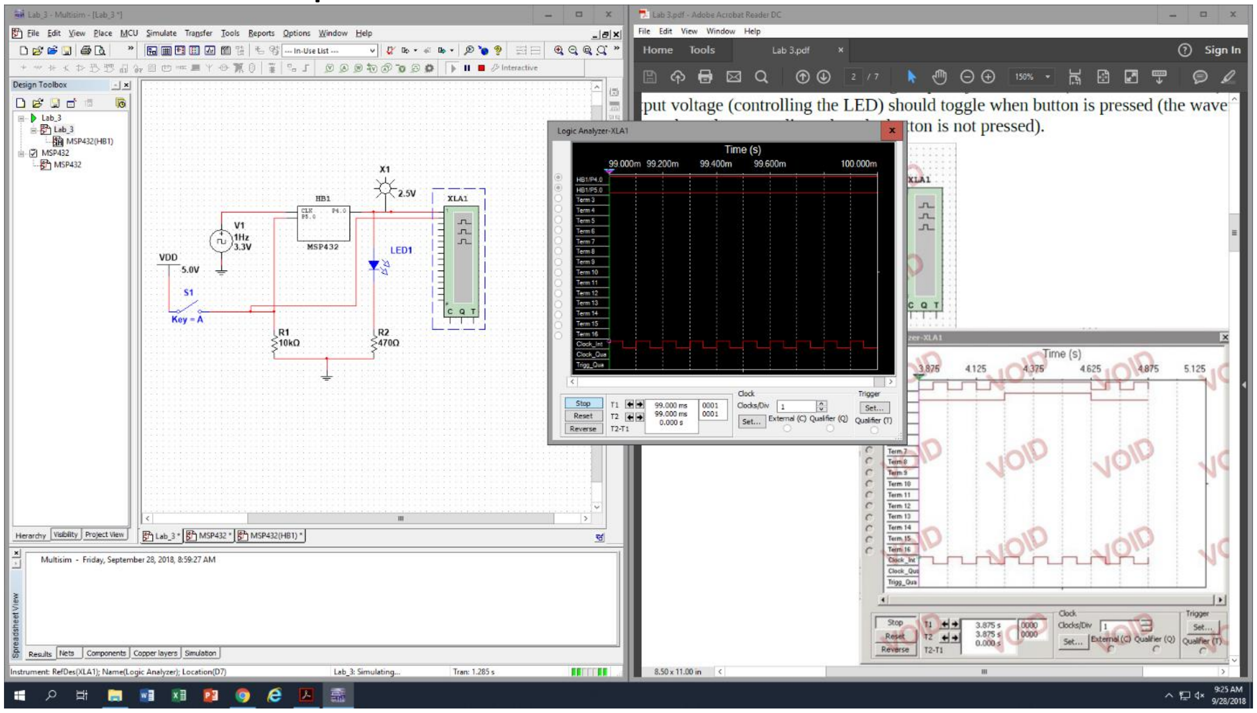
Introduction:

During this lab, we will be working with Code Composer Studio and will be looking at interfacing with the MSP432 board to control a red LED connected with a breadboard. First, we will be testing the design using Multisim before implementing it on the breadboard. Next, we will be running a loop to check for a button press and if pressed toggle the red LED using a delayed loop otherwise leave the LED on.

Procedure:

1. **Model and Simulate the External Hardware**
   - a) **Simulation**

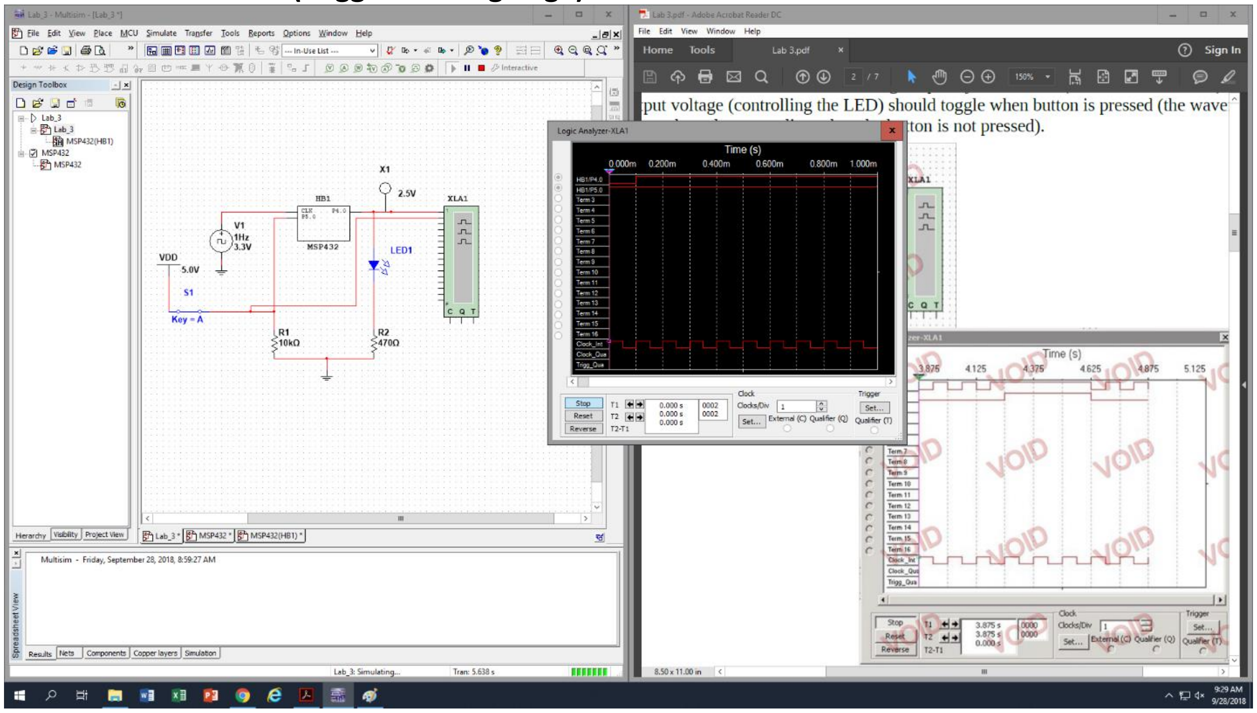
     Switch Open



put voltage (controlling the LED) should toggle when button is pressed (the wave ... tton is not pressed).

Multisim - Friday, September 28, 2018, 8:59:27 AM

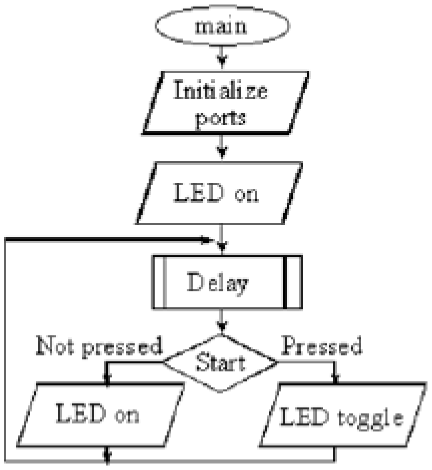
Switch Closed (Toggle at rising edge)


Lab_3 - Multisim - [Lab_3 *]
Design Toolbox
X1
2.5V
HB1
XLA1
V1
1Hz
3.3V
MSP432
LED1
VDD
5.0V
S1
Key = A
R1
10kΩ
R2
470Ω
Logic Analyzer-XLA1
Time (s)
Stop
Reset
Reverse
Multisim - Friday, September 28, 2018, 8:59:27 AM
Lab_3: Simulating...
Lab 3.pdf - Adobe Acrobat Reader DC
put voltage (controlling the LED) should toggle when button is pressed (the wave
ton is not pressed).
VOID
9:29 AM
9/28/2018

2. Modify ARM Assembly Program to Control the red LED

a) Pseudocode & Flowchart (to modify)

```
main    Initialize Port 1:
                Set the Port 1 direction register so P1.4 is an input and P1.0 is an output
                Set bit 4 of P1REN to enable the internal register for P1.4
                Specify this resistor as a pull up by setting bit 4 of P1OUT
        Set P1.0 so the LED is ON
loop    Delay about 100 ms
        Read the switch and test if the switch is pressed
        If P1.4=0 (the switch is pressed), toggle P1.0 (flip bit from 0 to 1, or from 1 to 0)
        If P1.4=1 (the switch is not pressed), set P1.0, so LED is ON
        Go to loop
```

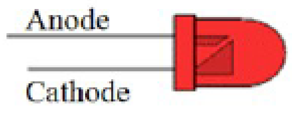


b) Revise Code: Change I/O Ports, Positive Logic, Longer Delay

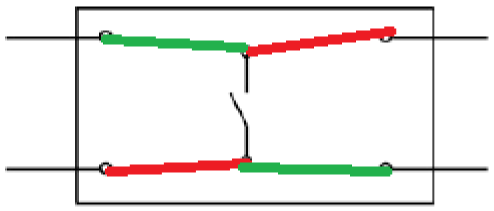
3. Assemble Hardware on Breadboard

a) Examine LED for anode/cathode

Anode is the positive side and has a longer tail than the cathode



b) Examine Switch for which pins are connected



Conclusion:

The result of this lab was determining how to interface with external circuits, how to use delays to make the LED blink, and to compartmentalize the code used. We made the code into snippets so that we may be able to reuse it when needed instead of having to re-write the code every time. This is beneficial as it reinforces the standard of Object Oriented Programming which is used in high level languages as well. We needed to implement positive logic by flipping the inputs using a not gate. finally, we had to add clock cycles to the delay to go from 100ms to 500 ms which is a 50% duty cycle.

Overall, this lab helped a lot with being able to split up larger segments of code to understand them better and explore the concept behind delays.

References:

For this project I used the lecture slides, the CCS tutorial, and the instruction set manual for this board.

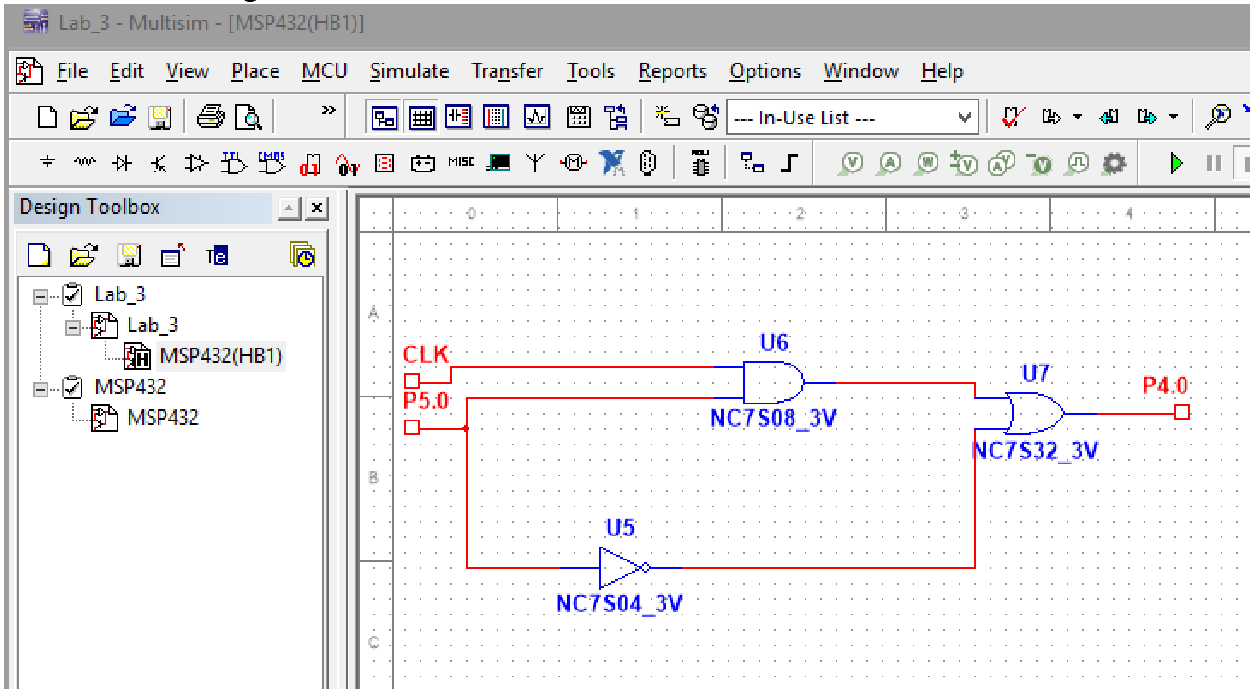
Appendix:

Pseudocode:

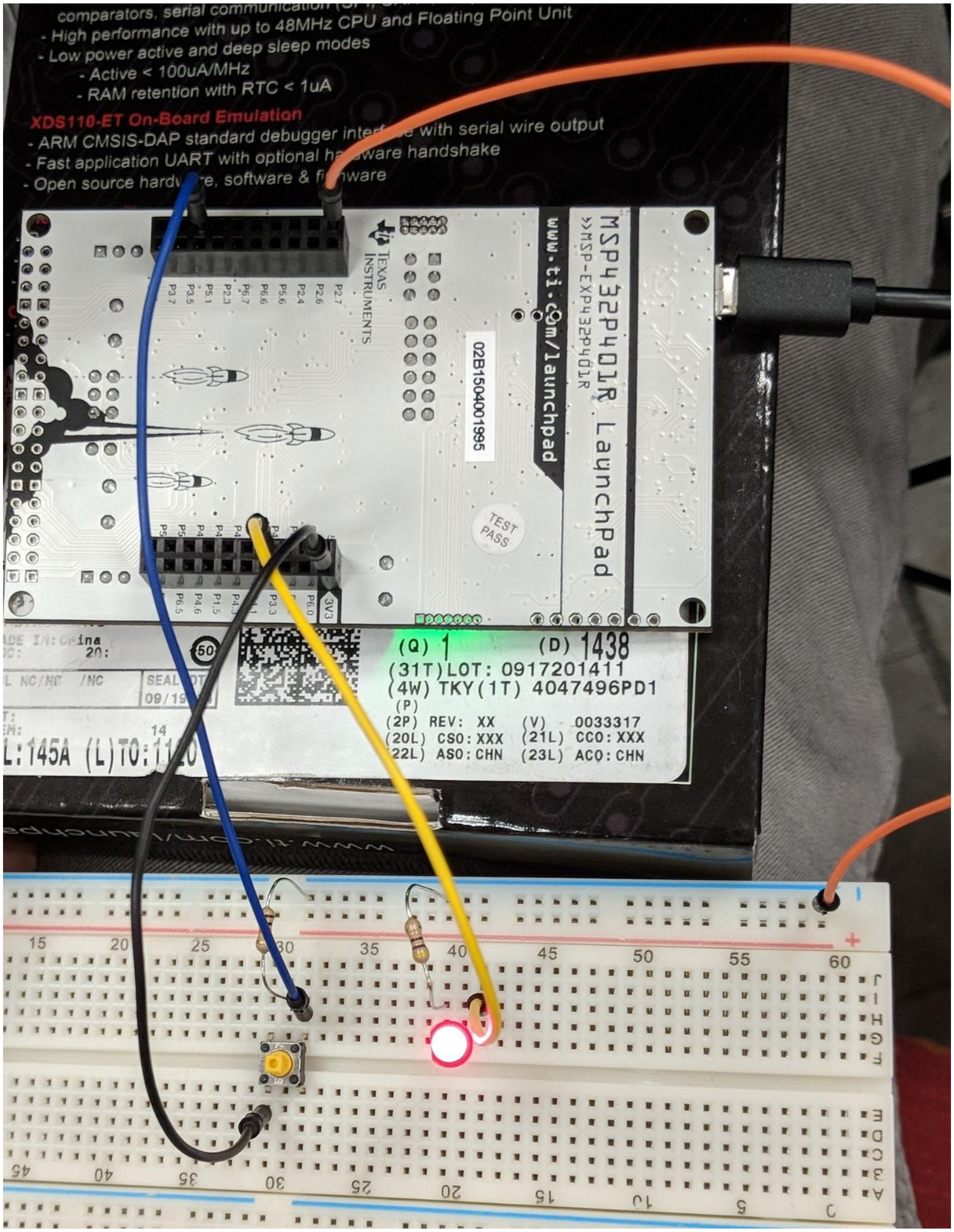
```
Initialize switch for input, and Port 5 for output to LED
Turn LED on for default

Start Loop
    Read input from switches
    Test is switch is pressed
        Switch 2
        None
        Catch Error if unexpected output
```

```
Set output
    Switch 2
        LED toggle
    None
        LED on
Restart Loop
```

Positive Logic:



Breadboard:


- High performance with up to 48MHz CPU and Floating Point Unit
- Low power active and deep sleep modes
- Active < 100uA/MHz
- RAM retention with RTC < 1uA
XDS110-ET On-Board Emulation
MSP432P401R LaunchPad
MSP-EXP432P401R
www.ti.com/launchpad
TEXAS INSTRUMENTS
TEST PASS
3V3

Code:

```
; InputOutput.s
; Runs on MSP432
; Test the GPIO initialization functions by setting the LED
; color according to the status of the switches.
; Daniel Valvano
; June 20, 2015

;  This example accompanies the book
;  "Embedded Systems: Introduction to the MSP432 Microcontroller",
;  ISBN: 978-1512185676, Jonathan Valvano, copyright (c) 2015
;  Section 4.2    Program 4.1
;
;Copyright 2015 by Jonathan W. Valvano, valvano@mail.utexas.edu
;    You may use, edit, run or distribute this file
;    as long as the above copyright notice remains
;THIS SOFTWARE IS PROVIDED "AS IS".  NO WARRANTIES, WHETHER EXPRESS, IMPLIED
;OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF
;MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE.
;VALVANO SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL,
;OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
;For more information about my classes, my research, and my books, see
;http://users.ece.utexas.edu/~valvano/

; built-in LED1 connected to P1.0
; negative logic built-in Button 1 connected to P1.1
; negative logic built-in Button 2 connected to P1.4
; built-in red LED connected to P2.0
; built-in green LED connected to P2.1
; built-in blue LED connected to P2.2
       .thumb

       .text
       .align  2
P4IN    .field 0x40004C21,32  ; Port 4 Input
P4OUT   .field 0x40004C23,32  ; Port 4 Output
P4DIR   .field 0x40004C25,32  ; Port 4 Direction
P4REN   .field 0x40004C27,32  ; Port 4 Resistor Enable
P4SEL0  .field 0x40004C2B,32  ; Port 4 Select 0
P4SEL1  .field 0x40004C2D,32  ; Port 4 Select 1
P5IN    .field 0x40004C40,32  ; Port 5 Input
P5OUT   .field 0x40004C42,32  ; Port 5 Output
P5DIR   .field 0x40004C44,32  ; Port 5 Direction
P5REN   .field 0x40004C46,32  ; Port 5 Resistor Enable
```

```
P5SEL0  .field 0x40004C4A,32  ; Port 5 Select 0
P5SEL1  .field 0x40004C4C,32  ; Port 5 Select 1

SW1      .equ 0x02              ; on the left side of the LaunchPad board
SW2      .equ 0x10              ; on the right side of the LaunchPad board

     .global main
     .thumbfunc main

main: .asmfunc
   BL  Port4_Init               ; initialize P1.1 and P1.4 and make them inputs (P1.1 and P1.4 built-in
buttons)
   BL  Port5_Init                                          ; initialize P5 for switch
loop
   BL  Port5_Input              ; read switch
   CMP R0, #0x01                ; R0 == 0x01?
   BEQ sw1pressed               ; if so, switch 1 pressed
          CMP R0, #0x00
          BEQ nopressed
   B   loop
sw1pressed
          BL  Delay                                                  ; delay 500ms
   BL  Port4_Output_Toggle        ; Toggle red led
   B   loop
nopressed
          BL  Port4_Output_On
          B   loop
   .endasmfunc
;------------Port4_Init------------
; Initialize GPIO Port 4 for negative logic switches on
; P4.0 as the LaunchPad is wired.
; Input: none
; Output: none
; Modifies: R0, R1
Port4_Init: .asmfunc
   ; configure P4.0 as GPIO
   LDR  R1, P4SEL0
   LDRB R0, [R1]
   BIC  R0, R0, #0x01           ; configure P4 as GPIO
   STRB R0, [R1]
   ; make P1.4 in
   LDR  R1, P4DIR
   LDRB R0, [R1]
   ORR  R0, R0, #0x01           ; output direction led
```

```
    STRB R0, [R1]
    BX   LR
    .endasmfunc
;------------Port5_Init------------
; Initialize GPIO Port 5 for positive logic switches on
; P5 as the LaunchPad is wired.
; Input: none
; Output: none
; Modifies: R0, R1
Port5_Init: .asmfunc
    ; configure P5 as GPIO
    LDR  R1, P5SEL0
    LDRB R0, [R1]
    BIC  R0, R0, #0x01             ; configure P5 as GPIO
    STRB R0, [R1]
    ; make P1.4 in
    LDR  R1, P5DIR
    LDRB R0, [R1]
    BIC  R0, R0, #0x01             ; input direction switch
    STRB R0, [R1]
    ; p5 in
    LDR  R1, P5IN
    LDRB R0, [R1]
    ORR  R0, R0, #0x00
    STRB R0, [R1]
    BX  LR
    .endasmfunc
;------------Port4_Input---------------
; Read and return the status of the switches.
; Input: P5IN
; Output: R0 0x01 if Switch 2 is pressed
; Modifies: R0, R1
Port5_Input: .asmfunc
    LDR  R1, P5IN
    LDRB R0, [R1]                ; read all 8 bits of Port 5
    AND  R0, R0, #0x01             ; select the input pin P5.0
    BX   LR
    .endasmfunc
;------------Port4_Output_On------------
; Read input and turn on the red led.
; Input: P4OUT
; Output: R0
; Modifies: R1
Port4_Output_On: .asmfunc
```

```
        LDR  R1, P4OUT
  LDRB R0, [R1]              ; read all 8 bits of Port 4
  ORR  R0, R0, #0x01         ; turn on red led
  STRB R0, [R1]
  BX   LR
  .endasmfunc
;------------Port4_Output_Toggle--------
; Read inputs and toggle the red led.
; Input: P4OUT
; Output: R0
; Modifies: R1
Port4_Output_Toggle: .asmfunc
        LDR  R1, P4OUT
  LDRB R0, [R1]              ; read all 8 bits of Port 4
  EOR  R0, R0, #0x01         ; toggle red led
  STRB R0, [R1]
  BX   LR
  .endasmfunc
;------------Delay----------------------
; Delay 500ms
; Modifies: R3
Delay: .asmfunc
        MOV  R3, #53800                                 ; Set delay cycles for 500ms
wait
        SUBS R3, R3, #0x01                      ; Waste clock cycles to get higher delay
        ADD  R3, R3, #0x01
        SUBS R3, R3, #0x01
        ADD  R3, R3, #0x01
        SUBS R3, R3, #0x01
        ADD  R3, R3, #0x01
        SUBS R3, R3, #0x01
        BNE  wait
        BX   LR
  .endasmfunc
  .end
```