```
;****************** main.s ***************
; Program written by: Dhruv Sandesara djs3967;
; Date Created: 2/4/2017
; Last Modified: 2/11/2017
; Brief description of the program
;   The LED toggles at 8 Hz and a varying duty-cycle
; Hardware connections (External: One button and one LED)
;  PE1 is Button input  (1 means pressed, 0 means not pressed)
;  PE0 is LED output (1 activates external9 LED on protoboard)
;  PF4 is builtin button SW1 on Launchpad (Internal)
;        Negative Logic (0 means pressed, 1 means not pressed)
; Overall functionality of this system is to operate like this
;   1) Make PE0 an output and make PE1 and PF4 inputs.
;   2) The system starts with the the LED toggling at 8Hz,
;      which is 8 times per second with a duty-cycle of 20%.
;      Therefore, the LED is ON for (0.2*1/8)th of a second
;      and OFF for (0.8*1/8)th of a second.
;   3) When the button on (PE1) is pressed-and-released increase
;      the duty cycle by 20% (modulo 100%). Therefore for each
;      press-and-release the duty cycle changes from 20% to 40% to 60%
;      to 80% to 100%(ON) to 0%(Off) to 20% to 40% so on
;   4) Implement a "breathing LED" when SW1 (PF4) on the Launchpad is pressed:
;      a) Be creative and play around with what "breathing" means.
;         An example of "breathing" is most computers power LED in sleep mode
;         (e.g., https://www.youtube.com/watch?v=ZT6siXyIjvQ).
;      b) When (PF4) is released while in breathing mode, resume blinking at 8Hz.
;         The duty cycle can either match the most recent duty-
;         cycle or reset to 20%.
;      TIP: debugging the breathing LED algorithm and feel on the simulator is impossible.
; PortE device registers
GPIO_PORTE_DATA_R  EQU 0x400243FC
GPIO_PORTE_DIR_R   EQU 0x40024400
GPIO_PORTE_AFSEL_R EQU 0x40024420
GPIO_PORTE_DEN_R   EQU 0x4002451C
; PortF device registers
GPIO_PORTF_DATA_R  EQU 0x400253FC
GPIO_PORTF_DIR_R   EQU 0x40025400
GPIO_PORTF_AFSEL_R EQU 0x40025420
GPIO_PORTF_PUR_R   EQU 0x40025510
GPIO_PORTF_DEN_R   EQU 0x4002551C
NUMBER              EQU 160000
NUMBER2             EQU 4000

SYSCTL_RCGCGPIO_R  EQU 0x400FE608
       IMPORT  TExaS_Init
       AREA    |.text|, CODE, READONLY, ALIGN=2
       THUMB
       EXPORT  Start
Start
    LDR R1,=SYSCTL_RCGCGPIO_R; TURN CLOCK ON
    LDR R0,[R1]
    ORR R0,R0,#0X30 ;enable clocks for port E and F
    STR R0,[R1]

    NOP
    NOP ; WAIT TWO CYCLES

    LDR R1,=GPIO_PORTF_DIR_R; DIRECTIONS OF SWITCHES
    LDR R0,[R1];
    AND R0,#0X0F; PF4 is input so zero
    STR R0,[R1];

    LDR R1,=GPIO_PORTE_DIR_R; DIRECTIONS OF SWITCHES
    LDR R0,[R1];
    AND R0,#0XD; PIN1 IS 0 SO INPUT
    ORR R0,#0X1; PIN0 IS 1 SO OUTPUT
    STR R0,[R1];




    LDR R1,=GPIO_PORTF_AFSEL_R; ALTERNATE FUNCTION SHUTOFF
    LDR R0,[R1];
    AND R0,#0XEF; SHUT DOWN PORT F PIN 4
    STR R0,[R1];

    LDR R1,=GPIO_PORTE_AFSEL_R; ALTERNATE FUNCTION SHUTOFF
    LDR R0,[R1];
    AND R0,#0XC; SHUT DOWN PORT E PIN 0 AND 1
    STR R0,[R1];


    LDR R1,=GPIO_PORTF_PUR_R; PULLUP REGISTER
    LDR R0,[R1];
    ORR R0,#0X10; PIN4 IS 1 NEGATIVE LOGIC
    STR R0,[R1];

    LDR R1,=GPIO_PORTF_DEN_R; DIGITAL ENABLE
    LDR R0,[R1]
    ORR R0,#0X10 ; PORT F PIN 4 DIGITAL ENABLE
    STR R0,[R1]


    LDR R1,=GPIO_PORTE_DEN_R; DIGITAL ENABLE
    LDR R0,[R1]
    ORR R0,#0X3 ; PORT E PIN 0 AND 1 DIGITAL ENABLE
    STR R0,[R1]



 ; TExaS_Init sets bus clock at 80 MHz
      BL  TExaS_Init ; voltmeter, scope on PD3
      CPSIE  I    ; TExaS voltmeter, scope runs on interrupts
























    MOV R5,  #2;
    MOV R7,#0;
    MOV R8,#100;
    MOV R9,#1;




loop



BREATHING
    LDR R3,=GPIO_PORTF_DATA_R; READ THE STATUS OF PORT E
    LDR R0,[R3]
    AND R0,R0,#0X10 ;ISOLATE THE STATUS OF SWITCH
    LSR R0,#4;
    CMP R0,#1;
    BEQ NOT_BREATHING

    CMP R7,#0;
    BNE NOT_AT_0;
    MOV R9,#1;     R7 IS AT 0 AND THEREFORE MAKE R9 INCREASING

NOT_AT_0
    CMP R7,#100;
    BNE NOT_AT_100
    MOV R9,#0;  R7 IS AT 100 AND THEREFORE MAKE R9 DECREASING

NOT_AT_100

    CMP R9,#0
    BEQ DECREASING

    ADD R7,R7,#1;
    B BREATHINGSTART


DECREASING
    SUB R7,R7,#1
















BREATHINGSTART

    MOV R1,R7

DELAY2
      CMP R1,#0
      BEQ DONE2
      LDR R0,=NUMBER2;
WAIT2  SUBS R0, #1;
      BNE WAIT2;  WAITING FOR THE 1 MS WHICH TAKES TO COUNT THE 3200
      SUBS R1,#1;   GETTING HOW MANY MS TO DELAY IN R1
      BNE DELAY2;
DONE2 B LEDOFF




LEDOFF  LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
    LDR R0,[R3]
    AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
    AND R0, R0, #0X0; OFF THE LED AND WRITE IT BACK
    STR R0,[R3]





    SUBS R1, R8,R7;


DELAY3
      CMP R1,#0
      BEQ DONE3
      LDR R0,=NUMBER2;
WAIT3  SUBS R0, #1;
      BNE WAIT3;  WAITING FOR THE 1 MS WHICH TAKES TO COUNT THE 3200
      SUBS R1,#1;   GETTING HOW MANY MS TO DELAY IN R1
      BNE DELAY3;
DONE3 B LEDON;

LEDON

    LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
    LDR R0,[R3]
    AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
    ORR R0, R0, #0X1; ON THE LED AND WRITE IT BACK
    STR R0,[R3]

    B BREATHING












































NOT_BREATHING

    LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF PORT E
    LDR R0,[R3]
    AND R0,R0,#0X2  ;ISOLATE THE STATUS OF SWITCH
    LSR R0,#1;
    CMP R0,#0;
    BEQ NOT_PRESSED;

PRESSED
    LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF PORT E
    LDR R0,[R3]
    AND R0,R0,#0X2  ;ISOLATE THE STATUS OF SWITCH
    LSR R0,#1;
    CMP R0,#1;
    BEQ PRESSED

    MOV R6,#10;
    SUB R6,R6,R5;
    CMP R6,#0;
    BEQ DUTYIS10

    ADD R5,R5,#2
    B NOT_PRESSED;

DUTYIS10
    MOV R5,#0;
    B NOT_PRESSED;
























NOT_PRESSED




    MOV R1,R5;

    BL DELAY;

    LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
    LDR R0,[R3]
    AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
    AND R0, R0, #0X0; OFF THE LED AND WRITE IT BACK
    STR R0,[R3]

    MOV R2,#10
    SUBS R1,R2,R5;
    BL DELAY;

    LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
    LDR R0,[R3]
    AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
    ORR R0, R0, #0X1; ON THE LED AND WRITE IT BACK
    STR R0,[R3]
```

```



      B    loop



DELAY
      CMP R1,#0
      BEQ DONE
      LDR R0,=NUMBER;
WAIT  SUBS R0, #1;
        BNE WAIT;  WAITING FOR THE 1 MS WHICH TAKES TO COUNT THE 3200
      SUBS R1,#1;   GETTING HOW MANY MS TO DELAY IN R1
      BNE DELAY
DONE      BX LR;






      ALIGN      ; make sure the end of this section is aligned
      END        ; end of file


```