```
;****************** 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
```

```
73        LDR R0,[R1];
74        AND R0,#0XEF; SHUT DOWN PORT F PIN 4
75        STR R0,[R1];
76
77        LDR R1,=GPIO_PORTE_AFSEL_R; ALTERNATE FUNCTION SHUTOFF
78        LDR R0,[R1];
79        AND R0,#0XC; SHUT DOWN PORT E PIN 0 AND 1
80        STR R0,[R1];
81
82
83        LDR R1,=GPIO_PORTF_PUR_R; PULLUP REGISTER
84        LDR R0,[R1];
85        ORR R0,#0X10; PIN4 IS 1 NEGATIVE LOGIC
86        STR R0,[R1];
87
88        LDR R1,=GPIO_PORTF_DEN_R; DIGITAL ENABLE
89        LDR R0,[R1]
90        ORR R0,#0X10 ; PORT F PIN 4 DIGITAL ENABLE
91        STR R0,[R1]
92
93
94        LDR R1,=GPIO_PORTE_DEN_R; DIGITAL ENABLE
95        LDR R0,[R1]
96        ORR R0,#0X3 ; PORT E PIN 0 AND 1 DIGITAL ENABLE
97        STR R0,[R1]
98
99
100
101     ; TExaS_Init sets bus clock at 80 MHz
102         BL  TExaS_Init ; voltmeter, scope on PD3
103         CPSIE  I    ; TExaS voltmeter, scope runs on interrupts
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128       MOV R5, #2;
129       MOV R7,#0;
130       MOV R8,#100;
131       MOV R9,#1;
132
133
134
135
136   loop
137
138
139
140   BREATHING
141       LDR R3,=GPIO_PORTF_DATA_R; READ THE STATUS OF PORT E
142       LDR R0,[R3]
143       AND R0,R0,#0X10 ;ISOLATE THE STATUS OF SWITCH
144       LSR R0,#4;
```

```
145       CMP R0,#1;
146       BEQ NOT_BREATHING
147
148       CMP R7,#0;
149       BNE NOT_AT_0;
150       MOV R9,#1;      R7 IS AT 0 AND THEREFORE MAKE R9 INCREASING
151
152   NOT_AT_0
153       CMP R7,#100;
154       BNE NOT_AT_100
155       MOV R9,#0;  R7 IS AT 100 AND THEREFORE MAKE R9 DECREASING
156
157   NOT_AT_100
158
159       CMP R9,#0
160       BEQ DECREASING
161
162       ADD R7,R7,#1;
163       B BREATHINGSTART
164
165
166   DECREASING
167       SUB R7,R7,#1
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184   BREATHINGSTART
185
186       MOV R1,R7
187
188   DELAY2
189         CMP R1,#0
190         BEQ DONE2
191         LDR R0,=NUMBER2;
192   WAIT2  SUBS R0, #1;
193         BNE WAIT2;  WAITING FOR THE 1 MS WHICH TAKES TO COUNT THE 3200
194         SUBS R1,#1;   GETTING HOW MANY MS TO DELAY IN R1
195         BNE DELAY2;
196   DONE2 B LEDOFF
197
198
199
200
201   LEDOFF  LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
202       LDR R0,[R3]
203       AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
204       AND R0, R0, #0X0; OFF THE LED AND WRITE IT BACK
205       STR R0,[R3]
206
207
208
209
210
211       SUBS R1, R8,R7;
212
213
214   DELAY3
215         CMP R1,#0
216         BEQ DONE3
```

```
217         LDR R0,=NUMBER2;
218   WAIT3  SUBS R0, #1;
219         BNE WAIT3;  WAITING FOR THE 1 MS WHICH TAKES TO COUNT THE 3200
220         SUBS R1,#1;   GETTING HOW MANY MS TO DELAY IN R1
221         BNE DELAY3;
222   DONE3 B LEDON;
223
224   LEDON
225
226       LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
227       LDR R0,[R3]
228       AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
229       ORR R0, R0, #0X1; ON THE LED AND WRITE IT BACK
230       STR R0,[R3]
231
232       B BREATHING
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277   NOT_BREATHING
278
279       LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF PORT E
280       LDR R0,[R3]
281       AND R0,R0,#0X2  ;ISOLATE THE STATUS OF SWITCH
282       LSR R0,#1;
283       CMP R0,#0;
284       BEQ NOT_PRESSED;
285
286   PRESSED
287       LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF PORT E
288       LDR R0,[R3]
```

```
289       AND R0,R0,#0X2  ;ISOLATE THE STATUS OF SWITCH
290       LSR R0,#1;
291       CMP R0,#1;
292       BEQ PRESSED
293
294       MOV R6,#10;
295       SUB R6,R6,R5;
296       CMP R6,#0;
297       BEQ DUTYIS10
298
299       ADD R5,R5,#2
300       B NOT_PRESSED;
301
302   DUTYIS10
303       MOV R5,#0;
304       B NOT_PRESSED;
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329   NOT_PRESSED
330
331
332
333
334       MOV R1,R5;
335
336       BL DELAY;
337
338       LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
339       LDR R0,[R3]
340       AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
341       AND R0, R0, #0X0; OFF THE LED AND WRITE IT BACK
342       STR R0,[R3]
343
344       MOV R2,#10
345       SUBS R1,R2,R5;
346       BL DELAY;
347
348       LDR R3,=GPIO_PORTE_DATA_R; READ THE STATUS OF THE LED
349       LDR R0,[R3]
350       AND R0,R0,#0X1  ;ISOLATE THE STATUS OF LED
351       ORR R0, R0, #0X1; ON THE LED AND WRITE IT BACK
352       STR R0,[R3]
353
354
355
356
357
358
359
360
```

```
361
362
363
364         B    loop
365
366
367
368   DELAY
369         CMP R1,#0
370         BEQ DONE
371         LDR R0,=NUMBER;
372   WAIT  SUBS R0, #1;
373           BNE WAIT;  WAITING FOR THE 1 MS WHICH TAKES TO COUNT THE 3200
374         SUBS R1,#1;   GETTING HOW MANY MS TO DELAY IN R1
375         BNE DELAY
376   DONE      BX LR;
377
378
379
380
381
382
383         ALIGN      ; make sure the end of this section is aligned
384         END        ; end of file
385
386
```