```
 1      .global EXTI0_IRQHandler
 2      .type EXTI0_IRQHandler, %function // needed for vector table
 3      .global main
 4
 5      .syntax unified
 6      .section .text.ButtonDemo
 7
 8  EXTI0_IRQHandler:
 9      push {lr}
10      //increment PHASE
11      ldr r0,=PHASE
12      ldr r1,[r0]
13      cmp r1,#2
14      beq ResetPhase
15      add r1, #1
16      b StorePhase
17  ResetPhase:
18      mov r1, #0
19  StorePhase:
20      str r1,[r0]
21      //small delay for debounce
22      mov r0,#0x00080000 // 10 ms
23  Bloop:
24      subs r0,#1
25      bne Bloop
26
27      //exit the handler
28      bl Reset_EXTI0
29      pop {lr}
30      bx r14
31
32  main:
33      bl InitButton //initialize PA0 as input from button
34      bl Init_EXTI0 //button to trigger EXTI0
35      bl InitLEDs //LEDs on PB9-8-7-6
36
37      cpsie i //enable interrupts
38  loop:
39      bl PhaseDisplay
40      b loop
41
42      //endless loop of incrementing MCount every half second
43
44
45      .data
46  PHASE:  .word 0
47  COUNTER: .word 0
48          .global COUNTER
49          .global PHASE
50
51      .end
52
```

```
 1      .include "Equates.s"
 2
 3      .global InitButton
 4      .global CheckButton
 5      .global Init_EXTI0
 6      .global Reset_EXTI0
 7
 8      .syntax unified
 9      .section   .text.ButtonDriver
10
11  // GPIO initialization for button
12  InitButton:
13      ldr r0,=RCC //RCC register block
14      ldr r1,[r0,#AHBENR] //read RCC_AHB1ENR
15      orr r1,#GPIOAEN // enable GPIOA clock
16      str r1,[r0,#AHBENR] // update AHB1ENR
17      ldr r0,=GPIOA //GPIOA register block
18      ldr r1,[r0,#MODER] //current mode register
19      bic r1,#0x03 //MODER[1:0] = 00 for PA0 input
20      str r1,[r0,#MODER] //update mode register
21      bx lr //return
22
23  // CheckButton - return state of push button
24  // r0 = return value of 0 or 1
25  CheckButton:
26      ldr r0,=GPIOA //GPIO port A
27      ldrh r0,[r0,#IDR] //set bit
28      and r0,#0x01 //mask all but bit 0
29      bx r14 //return
30
31  Init_EXTI0:
32      //select PA0 as EXTI0
33      ldr r1,=SYSCFG
34      ldrh r2,[r1,#EXTICR1] //EXTI priorities for EXTI0
35      bic r2,#0x0f          //bits 3-0 = 0000 to select PA0 = EXTI0
36      strh r2,[r1,#EXTICR1] //EXTI priorities for EXTI0
37      //configure EXTI0 as rising edge triggered
38      ldr r1,=EXTI
39      mov r2,#1             //bit #0 for EXTI0
40      str r2,[r1,#FTSR]     //select falling edge trigger
41      str r2,[r1,#PR]       //clear any pending event
42      str r2,[r1,#IMR]      //enable EXTI0
43      //configure NVIC to enable EXTI0 as priority 1
44      ldr r1,=NVIC_ISER0
45      mov r2,#0x40          //EXTI0 is IRQ 6
46      str r2,[r1]           //Set enable IRQ 6
47      ldr r1,=NVIC_IPR1
48      mov r2,#0x00100000    //Make EXTI0 priority 1
49      str r2,[r1]           //Write IPR1 3rd byte
50      bx lr
51
52  Reset_EXTI0:
53      // Reset EXTI0 pending bit in EXTI
54      ldr r0,=EXTI //point to EXTI registers
55      mov r1,#0x01 //bit 0 = EXTI0 pending bit
56      str r1,[r0,#PR] //reset EXTI0 pending bit (write 1 to it)
57      // Reset EXTI0 pending bit in NVIC (in case triggered by bounce)
```

```
58      ldr r0,=NVIC_ICPR0 //clear Interrupt Pending Register
59      mov r1,#0x40 //EXTI0 = bit 6 of that register
60      str r1,[r0]
61      bx lr
62
63      .end
64
65
```

```
 1     .include "Equates.s"
 2
 3     .global InitLEDs    //init GPIOB9-6 for LEDs
 4     .global DisplayNum //display 4-bit # on LEDs
 5     .global PhaseDisplay
 6
 7     .syntax unified
 8     .section    .text.LEDdrivers
 9
10 // GPIOB initialization for LEDs: PB9-8-7-6
11 InitLEDs:
12     ldr r0,=RCC         //RCC register block
13     ldr r1,[r0,#AHBENR] //read RCC_AHB1ENR
14     orr r1,#GPIOBEN     //enable GPIOB clock
15     str r1,[r0,#AHBENR] //update AH1ENR
16     ldr r0,=GPIOB       //GPIOA register block
17     ldr r1,[r0,#MODER]  //current mode register
18     bic r1,#0x000FF000  ///MODER[19-12] = 00000000
19     orr r1,#0x00055000  //MODER[19-12] = 01010101
20     str r1,[r0,#MODER]  //update mode register
21     ldr r1,[r0,#ODR]    //output data register
22     bic r1,#0x03C0       //PB9-6 = 0000 (all LEDs off)
23     str r1,[r0,#ODR]    //update output data register
24     bx lr
25
26
27 ////---------------------PHASES-------------------------//
28 PhaseDisplay:
29     push {r1,r2,r3,r4,lr}
30     ldr r1,=PHASE
31     ldr r2,[r1]
32     cmp r2,#0
33     beq Phase0
34     cmp r2,#1
35     beq Phase1
36     cmp r2,#2
37     beq Phase2
38 //---------------------PHASE 0-------------------------//
39 Phase0:
40     ldr r2,=GPIOB
41     ldrh r3,[r2,#ODR]
42     mov r3,#0
43     strh r3, [r2, #ODR]
44 loop0:
45     ldr r1,=PHASE
46     ldr r2,[r1]
47     cmp r2,#0
48     beq loop0
49     pop {r1,r2,r3,r4,lr}
50     bx lr
51
52 //---------------------PHASE 1-------------------------//
53 Phase1:
54     ldr r1,=COUNTER
55     mov r2,#0
56     str r2,[r1]
57     mov r4, #0
```

```
 58
 59 Phase1Loop:
 60     ldr r3,=GPIOB
 61     ldrh r1,[r3,#ODR] //read ODR
 62     bic r1,#0x03C0 //clear bits 9-6
 63
 64     ldr r2,=COUNTER
 65     ldr r2,[r2]
 66     cmp r2,#0
 67     beq Red
 68     cmp r2,#1
 69     beq RedBlue
 70     cmp r2,#2
 71     beq RedBlueOrange
 72     cmp r2,#3
 73     beq AllOn1
 74     cmp r2,#4
 75     beq AllOff1
 76
 77 Red:
 78     mov r2,#1
 79     lsl r2,#6
 80     orr r1,r2
 81     strh r1,[r3,#ODR]
 82     bl Delay
 83     b CheckPhase1
 84
 85 RedBlue:
 86     mov r2,#3
 87     lsl r2,#6
 88     orr r1,r2
 89     strh r1,[r3,#ODR]
 90     bl Delay
 91     b CheckPhase1
 92
 93 RedBlueOrange:
 94     mov r2,#7
 95     lsl r2,#6
 96     orr r1,r2
 97     strh r1,[r3,#ODR]
 98     bl Delay
 99     b CheckPhase1
100
101 AllOn1:
102     bl AllOn
103     bl Delay
104     b CheckPhase1
105
106 AllOff1:
107     bl AllOff
108     bl Delay
109
110
111 CheckPhase1:
112     //check if PHASE is equal to 1, exit if not.
113     ldr r3,=PHASE
114     ldr r2,[r3]
```

```
115     cmp r2,#1
116     bne ExitPhase1
117     bl IncrementCounter
118     b Phase1Loop   //Go back to Phase1 to show next LEDs
119
120 ExitPhase1:
121     pop {r1,r2,r3,r4,lr}
122     bx lr
123
124 //---------------------PHASE 2-------------------------//
125 Phase2:
126     ldr r1,=COUNTER
127     mov r2,#0
128     str r2,[r1]
129     mov r4, #1
130
131 Phase2Loop:
132     ldr r3,=GPIOB
133     ldrh r1,[r3,#ODR] //read ODR
134     bic r1,#0x03C0 //clear bits 9-6
135
136     ldr r2,=COUNTER
137     ldr r2,[r2]
138     cmp r2,#0
139     beq Green
140     cmp r2,#1
141     beq GreenOrange
142     cmp r2,#2
143     beq GreenOrangeBlue
144     cmp r2,#3
145     beq AllOn2
146     cmp r2,#4
147     beq AllOff2
148
149 Green:
150     mov r2,#8
151     lsl r2,#6
152     orr r1,r2
153     strh r1,[r3,#ODR]
154     bl Delay
155     b CheckPhase2
156
157
158 GreenOrange:
159     mov r2,#0xC
160     lsl r2,#6
161     orr r1,r2
162     strh r1,[r3,#ODR]
163     bl Delay
164     b CheckPhase2
165
166 GreenOrangeBlue:
167     mov r2,#0xE
168     lsl r2,#6
169     orr r1,r2
170     strh r1,[r3,#ODR]
171     bl Delay
```

```
172     b CheckPhase2
173
174 AllOn2:
175     bl AllOn
176     bl Delay
177     b CheckPhase2
178
179 AllOff2:
180     bl AllOff
181     bl Delay
182
183
184 CheckPhase2:
185     //check if PHASE is equal to 2, exit if not.
186     ldr r3,=PHASE
187     ldr r2,[r3]
188     cmp r2,#2
189     bne ExitPhase2
190
191     bl IncrementCounter
192     b Phase2Loop  //Go back to Phase1 to show next LEDs
193
194 ExitPhase2:
195     pop {r1,r2,r3,r4,lr}
196     bx lr
197
198 //-------LEDs On/Off--------//
199 AllOn:
200     push {r1,r2,lr}
201     mov r2,#0xF
202     lsl r2,#6
203     orr r1,r2
204     strh r1,[r3,#ODR]
205     pop {r1,r2,lr}
206     bx lr
207
208 AllOff:
209     push {r1,r2,lr}
210     mov r2,#0
211     strh r2, [r3, #ODR]
212     pop {r1,r2,lr}
213     bx lr
214
215 //------COUNTER------//
216 IncrementCounter:
217     //increment phase counter to change phase 1 pattern.
218     push {r1,r2,lr}
219     ldr r1,=COUNTER
220     ldr r2,[r1]
221     cmp r2,#4 //0-4
222     beq resetCounter
223     add r2,#1
224     str r2,[r1]
225     b exitCounter
226 resetCounter:
227     mov r2,#0
228     str r2,[r1]
```

```
229 exitCounter:
230     pop {r1,r2,lr}
231     bx lr
232 
233 //--------------DELAY----------//
234 Delay:
235     push {r1,r4,lr}
236     cmp r4,#1
237     beq Phase2Delay
238     ldr r1,=0x00200000 //delay count for .5 second
239     b Dloop
240 Phase2Delay:
241     ldr r1,=0x00400000 //delay count for 1 second
242 Dloop:
243     //decrement delay count
244     subs r1,#1
245     bne Dloop //repeat if not r1 not equal to 0
246     pop {r1,r4,lr}
247     bx lr
248 
249     .end
250 
```