```
 1     .include "Equates.s"
 2     .global HalfSecs
 3
 4     .syntax unified
 5     .section    .text.Timer_App
 6 //--------PATTERN FUNCTION---------//
 7 SetPattern:
 8     push {lr}
 9     ldr r1,=PATTERN
10     ldr r2,[r1]
11     cmp r2,#1
12     beq RedBlue
13     cmp r2,#3
14     beq RedBlueOrange
15     cmp r2,#7
16     beq AllOn
17     cmp r2,#0xF
18     beq AllOff
19 Red:
20     mov r2,#1
21     str r2,[r1]
22     b Skip
23 RedBlue:
24     mov r2,#3
25     str r2,[r1]
26     b Skip
27 RedBlueOrange:
28     mov r2,#7
29     str r2,[r1]
30     b Skip
31 AllOn:
32     mov r2,#0xF
33     str r2,[r1]
34     b Skip
35 AllOff:
36     mov r2,#0
37     str r2,[r1]
38 Skip:
39     pop {lr}
40     bx lr
41
42 //---------DELAY---------//
43
44 Delay:
45     ldr r1,=0x00300000
46 Dloop1:
47     subs r1,#1
48     bne Dloop1
49     subs r0,#1
50     bne Delay
51     bx lr
52
53 //EXTI0 IRQHandler for PHASE
54     .global EXTI0_IRQHandler
55     .type EXTI0_IRQHandler, %function // needed for vector table
56 EXTI0_IRQHandler:
57     push {lr}
```

```
 58     //Switch PHASE
 59     ldr r0,=PHASE
 60     ldr r1,[r0]
 61     eor r1,#1
 62     str r1,[r0]
 63     //Reset pattern to 0
 64     ldr r0,=PATTERN
 65     mov r1,#0
 66     str r1,[r0]
 67     //small delay for debounce
 68     mov r0,#0x00080000 // 10 ms
 69 Bloop:
 70     subs r0,#1
 71     bne Bloop
 72     //exit the handler
 73     bl Reset_EXTI0
 74     pop {lr}
 75     bx r14
 76
 77 //TIM6 interrupt handler for LED timer.
 78     .global TIM6_DAC1_IRQHandler
 79     .type TIM6_DAC1_IRQHandler, %function
 80 TIM6_DAC1_IRQHandler:
 81     push {lr}
 82     bl TIM6_ResetUIF    //Reset UIF flag of TIM6
 83     ldr r0,=PHASE
 84     ldr r1,[r0]
 85     cmp r1,#0
 86     beq SkipPattern
 87     bl SetPattern
 88 SkipPattern:
 89     bl PhaseDisplay
 90     ldr r0,=HalfSecs //Increment sample #
 91     ldr r1,[r0] //current sample#
 92     add r1,#1 //increment
 93     str r1,[r0] //save sample#
 94     cmp r1,#4
 95     blo Tskip6 //skip next if < 4
 96     mov r1,#0   //reset counter
 97     str r1,[r0]
 98     bl PhaseDisplay
 99 Tskip6:
100     pop {lr}
101     bx lr
102
103 // MAIN
104         .global main
105 main:
106     bl InitButton //initialize PA0 as input from button
107     bl Init_EXTI0 //button to trigger EXTI0
108     bl InitLEDs
109     bl TIM6_Init
110     mov r1,#0
111     ldr r0,=HalfSecs
112     cpsie i
113 Mloop:
114     mov r0,#1   //1 half second
```

```
115     bl Delay
116
117     ldr r4,=Count
118     ldr r5,[r4]
119     add r5,#1
120     str r5,[r4]
121
122     //bl PhaseDisplay
123     b Mloop
124
125                 .data
126 HalfSecs:      .word  0
127 PATTERN:       .word  0
128 PHASE:         .word  0
129 Count:         .word  0
130                 .global PATTERN
131                 .global PHASE
132
133     .end
134
135
```

```
 1      .include "Equates.s"
 2      .global TIM6_Init
 3      .global TIM6_ResetUIF
 4      .global TIM6_Start
 5      .global TIM6_Stop
 6
 7      .equ PreScale, 9999
 8      .equ MinARR,    399
 9
10      .syntax unified
11      .section    .text.TimerDriver
12
13  // Initialize TIM6 to interrupt every 1/2 second
14  TIM6_Init:
15      //enable the clock to TIM6
16      ldr r0,=RCC             //Clock control
17      ldr r1,[r0,#APB1ENR]    //APB1 clock enable bits
18      orr r1,#TIM6EN          //Enable TIM6 clock
19      str r1,[r0,#APB1ENR]    //update APB1
20      //configure TIM6 for a 0.5 Hz event period, given CK_INT=8MHz
21      ldr r0,=TIM6            //TIM6 registers
22      mov r1,#PreScale        //Prescale for waveform generation
23      str r1,[r0,#PSC]
24      mov r1,#MinARR          //Auto-repeat for waveform generation
25      str r1,[r0,#ARR]
26      mov r1,#0x0001          //Counter Enable (CEN)
27      str r1,[r0,#CR1]
28      mov r1,#0x0001          //Update Interrupt Enable (UIE)
29      str r1,[r0,#DIER]
30      //Enable TIM6 interrupt in NVIC
31      ldr r0,=NVIC_ISER0      //NVIC_ISER1 enable registers
32      mov r1,#1
33      lsl r1,#TIM6_BIT        //Shift 1 to TIM6 enable bit in ISER1
34      str r1,[r0,#TIM6_OFF]   //enable TIM6 interrupt in ISER1
35      bx lr
36
37  //--------------
38  //Reset UIF flag in the TIM6 status register
39  TIM6_ResetUIF:
40      push {r0,r1}
41      ldr r0,=TIM6
42      ldr r1,[r0,#SR]         //Read SR
43      bic r1,#0x0001          //Clear UIF (flag)
44      str r1,[r0,#SR]         //Update Sr
45      pop {r0,r1}
46      bx lr
47
48  //-------------
49  // Start TIM6
50  TIM6_Start:
51      ldr r0,=TIM6            //TIM6 registers
52      mov r1,#0x0001          //Counter Enable (CEN)
53      str r1,[r0,#CR1]        //CEN=1
54      bx lr
55
56  //-------------
57  // Stop TIM6
```

```
58 TIM6_Stop:
59     ldr r0,=TIM6
60     mov r1,#0x0000
61     str r1,[r0,#CR1]
62     bx lr
63
64     .end
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 ////---------------------PHASES--------------------------//
27 PhaseDisplay:
28     push {r1,r2,r3,r4,lr}
29     ldr r1,=PHASE
30     ldr r2,[r1]
31     cmp r2,#0
32     beq Phase0
33     cmp r2,#1
34     beq Phase1
35 //---------------------PHASE 0--------------------------//
36 Phase0:
37     mov r2,#0
38     bl DisplayNum
39     pop {r1,r2,r3,r4,lr}
40     bx lr
41 //---------------------PHASE 1--------------------------//
42 Phase1:
43     ldr r1,=PATTERN
44     ldr r2,[r1]
45     bl DisplayNum
46     pop {r1,r2,r3,r4,lr}
47     bx lr
48
49 //---------------------------------//
50
51 DisplayNum:
52     push {r1,r2,r3}
53     ldr r3,=GPIOB
54     ldrh r1,[r3,#ODR]
55     bic r1,#0x03C0
56     lsl r2,#6
57     orr r1,r2
```

```
58      strh r1,[r3,#ODR]
59      pop {r1,r2,r3}
60      bx lr
61
62      .end
63
```

```
 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
```