ECE 375
Computer Organization and Assembly Language Programming
Fall 2021
Homework #3

[25 pts]

1- Consider the SHORTDELAY subroutine that is outlined below. Write AVR assembly so that it utilizes the 16-bit Timer/Counter1 to delay for 201 milliseconds (0.201 seconds). Assume that the system clock frequency is 16 MHz. Note the following requirements:

(a) Timer/Counter1 must be initialized to operate in the Normal mode.

(b) The SHORTDELAY subroutine loads the proper value into TCNT1 and waits until TOV1 is set. Once TOV is set, it is cleared and the SHORTDELAY subroutine returns.

(c) The code must utilize a polling strategy to check TOV1 (do not use interrupt vectors in this homework problem).

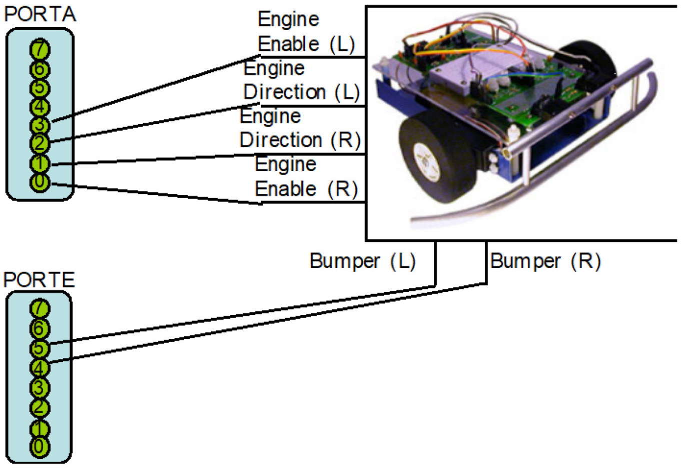
Use the skeleton code shown below. Also, show the necessary calculations for determining TCNT1 and the best prescaler value. Your code must not use any other GPRs besides mpr. Please don't hesitate to refer to the ATmega128 datasheet.

```
.include "m128def.inc"
.def mpr = r16
...
.ORG   $0000
       RJMP Initialize
.ORG   $0046                   ; End of interrupt vectors
Initialize:
       ...
       ...Your code goes here...
       ...
SHORTDELAY:
       ...
       ...Your code goes here...
       ...
       RET
```

[25 pts]

2- Imagine that you want to program your lab board to handle the I/O configuration illustrated in the image below. Review the starter code that is provided below and fill in the missing lines (based on the instructions) to accomplish the given tasks. mpr is the only general purpose register that you are allowed to use in the code.

```
.include “m128def.inc”
.def mpr = r16
.org $0000
     rjmp      INIT
.org _______________(i)
     rjmp HitRight
.org _______________(ii)
     rjmp HitLeft
     …
.org $0046
INIT: _______________________(1)       ; Configure direction of engine pins
     ________________________(2)       ;
     ________________________(3)       ; Configure direction of bumper pins
     ________________________(4)       ;
     ________________________(5)       ; Enable pull-up resisters
     ________________________(6)       ;          for L/R bumpers
     ________________________(7)       ; Detect on the proper edge
     ________________________(8)       ;
     ________________________(9)       ; Turn on interrupts for L/R bumpers
     ________________________(10)      ;
     sei                               ; Turn on global interrupt
```



(a) Fill in lines 1-2 so that the pin directions are properly configured to control the engine enable and engine direction for both left and right wheels. Any unused pins must be configured as inputs.

(b) Fill in lines 3-4 so that the pin directions are properly configured to detect left and right bumper movements. Once again, any unused pins must be configured as inputs.

(c) What are the addresses needed in lines (i) and (ii) to properly control the execution of interrupt service routines for left and right bumpers? Fill in lines 5-6 to enable the pull-up resisters for these whiskers.

(d) Fill in lines 7-8 with the necessary code to set External Input Sense Control to detect bumper hits (i.e., interrupts) on a falling edge.

(e) Fill in lines 9-10 to unmask the interrupts for whisker movements.

[25 pts]

3- Consider the AVR code segment shown below (with some missing information) that configures Timer/Counter0 for Fast PWM operation, and modifies the Fast PWM duty cycle whenever a specific button on Port D is pressed.

(a) Fill in lines (1-2) with the instructions necessary to configure Timer/Counter0 for Fast PWM mode, non-inverting output, and a prescale value of 8.

(b) Based on the prescale value used in part (a), what is the frequency of the PWM signal (f_{PWM}) being generated by Timer/Counter0? Assume the system clock frequency is 16 MHz.

(c) Fill in lines (3-4) to provide the compare value for Timer/Counter0 so that the initial duty cycle is 51% (use the closest available settings).

(d) What would be the value necessary for the variable `step` to increase the duty cycle by 12.5% each time the `DUTY_STEP` subroutine is executed? Ignore the case when/if the compare value overflows.

```
.include   "m128def.inc"
.def       mpr = r16
.def       temp = r17
.equ       step = ___

INIT:
     …
     ; stack pointer is initialized
     …

     ; I/O ports
     ldi   mpr, 0b00010000   ; set pin 4 (OC0) as output
     out   DDRB, mpr

     ldi   mpr, 0b00000000   ; set pin 0 as input
     out   DDRD, mpr
     ldi   mpr, 0b00000001   ; enable pull-up resistor for pin 0
     out   PORTD, mpr

     ; Timer/Counter0
     ; Fast PWM mode, non-inverting, prescale = 8
     __________________________(1)
     __________________________(2)

     ; Initial compare value for PWM output
     __________________________(3)
     __________________________(4)

MAIN:
     sbis  PIND, 0
     rcall      DUTY_STEP
     rjmp  MAIN

DUTY_STEP:
     push  mpr
     push  temp

     in    mpr, ________     ; read the current PWM compare value
     ldi   temp, step
     add   mpr, temp         ; add step value to compare value
     out   ________, mpr     ; write new PWM compare value

     pop   temp
     pop   mpr
     ret                     ; return
```