Tu Lam

ECE 375 / Dr. Justin Goins

Nov 11th, 2020

Homework #3

(Due Date: Nov 18th, 2020)

1. *The AVR code below (with some information missing) is designed to initialize and service interrupts from three I/O devices (DevA, DevB, and DevC).*

a) There are 8 external interrupt pins (INT0-INT7) in AVR. Based on the code provided, which three interrupt pins are these I/O devices connected to and what is the immediate value needed in line 1?

Answer: There are three interrupt pins that these I/O devices are connected to. We have ***Pin 4 & 5 (INT 2)*** from ***EICRA*** and ***Pin 2 (INT 5)*** from ***EICRB***. And the immediate value that needed to be in line 1 is ***0b01100100***. The interrupt ***INT 2 & 5*** will connect to the ***ISR_DevA & ISR_DevB*** and ***INT 6*** will be ***ISR_DevC***.

b) Which I/O device's interrupt is triggered by a low-level input?

Answer: The I/O device that get trigger by a low-level input is at interrupt $000E where the ***INT 6*** in ***EICRB*** is a low-level at ***pin 4 & 5***.

c) Assume that pins 4 and 3 on PORTA are connected to an additional output device (DevD). Fill in lines 2-3 such that the corresponding pins are specified as outputs. Do not configure any other pins on the port as outputs.

Answer: *ldi mpr, 0b00011000* (***Line 2***)
out DDRA, mpr (***Line 3***)

d) Suppose DevA requires that no interrupts are detected while it is being serviced. Fill in lines 4-5 with the necessary code to clear any latched interrupts at the end of ISR_DevA.

Answer: *ldi mpr, $FF* (***Line 4***)
out EIFR, mpr (***Line 5***)

e) Consider the RCALL instruction on line 0. Would the program function correctly if this line was replaced with the following underlined text? CALL ISR_DevB. Assume that no other lines of code are changed. Provide an explanation to support your answer.

Answer: If we were to change the RCALL instruction to CALL instruction, this function would not run correctly even though as both can do the same thing. Since one instruction is to call only with 1 word (16-bit) while the other one require 2 words (32-bit), they will still perform the same function as the flash usage in one instruction is less than other in term of saving program memory.

2. 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.

Answer: *ldi mpr, 0b01101010* (***Line 1***)
out TCCR0, mpr (***Line 2***)

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.

Answer: $f_{PWM} = \frac{CLK_{IO}}{prescale \times 256} \rightarrow \frac{16MHZ}{8 * 256} = 7.8\ cycle/sec$

c) Fill in lines (3-4) to provide the compare value for Timer/Counter0 so that the initial duty cycle is 0%.

Answer: *in mpr, OCR0* (***Line 3***)
cpi mpr, 255 (***Line 4***)

d) What would be the value necessary for the variable step to increase the duty cycle by 10% each time the DUTY_STEP subroutine is executed? Ignore the case when/if the compare value overflows.

Answer: To figure out the 10% duty cycle, we could take the maximum step in the 8-bit counter which is the value of 255 (256 if overflow), then dividing it by 10% and we get:

$$\frac{256}{10} = 25.6\ or\ 26$$

So the necessary value for the variable step to increase the duty cycle by 10% is the value of ***26***.