Jonathan W. Valvano First:_________________ Last:_____________________

October 11, 2018, 3:30-4:45pm. This is a closed book exam, with one 8.5 by 11-inch crib sheet. You have 75 minutes, so please allocate your time accordingly. ***Please read the entire quiz before starting***.

(5) Question 1. Consider the IoT system from Lab 4 that sends time information from the TM4C123 to the phone. For this problem, the goal is to display current time on the phone, where the time maintained by the TM4C123 interrupt service routine is defined as truth. The current time is periodically sent from the TM4C123 to the phone as a set of virtual pins. However, because the data are first sent to the ESP8266, then sent to the hot spot, then sent to the Blynk server, and finally sent to the phone, there will be a delay or lag between true time on the TM4C123 and displayed time on the phone. Regardless of this delay, the system has value, but the value of the system increases as this lag decreases. I.e., all things being equal, the system performance improves as the lag gets shorter. **Is this system real time?** If so, what type of real-time system is it? If not real time, explain why it is not real time. Put your answer in the box.

(5) Question 2. How do you use a voltmeter or a regular oscilloscope (not a spectrum analyzer) to **quantify the amount of noise** on the 3.3V power line?

(5) Question 3. In order to improve **signal to noise ratio** on data sampled by the ADC we can deploy *over-sampling*. Assume for this problem, hardware averaging is not activated. This means for every one output we need, we will sample the ADC N times. Which of the following operations will improve the signal to noise ratio over sampling just once? There may be multiple answers, if so list all operations that improve the signal to noise ratio.

A) Calculate the maximum of the N samples.
B) Calculate the average difference between successive samples.
C) Calculate the mean of the samples.
D) Calculate the minimum of the N samples.
E) Perform a linear regression and calculate the slope of the data.

(5) Question 4. One can use a **probability mass function** to describe noise on the ADC. Assume the input to the ADC is 1.65V, and therefore the expected ADC sample is 2048. For this question, **draw a pmf graph** illustrating noise typical of the TM4C123, and label the two axes.

(10) Question 5. Consider the following C code (with corresponding compiler generated assembly). Periodic SysTick interrupts occur every 1 ms.

`0x00000440 4806      LDR   r0,[pc,#24]`
`0x00000442 6800      LDR   r0,[r0,#0x00]`
`0x00000444 F0800004  EOR   r0,r0,#0x04`
`0x00000448 4905      LDR   r1,[pc,#20]`
`0x0000044A F8C103FC  STR   r0,[r1,#0x3FC]`
`0x0000044E 4805      LDR   r0,[pc,#20]`
`0x00000450 6800      LDR   r0,[r0,#0x00]`
`0x00000452 F1000001  ADD   r0,r0,#0x01`
`0x00000456 4903      LDR   r1,[pc,#12]`
`0x00000458 6008      STR   r0,[r1,#0x00]`
`0x0000045A 4770      BX    lr`
	`// Interrupt service routine`
	`void SysTick_Handler(void){`
	`  GPIO_PORTF_DATA_R ^= 0x04;`
	`  Counts = Counts + 1;`
	`}`

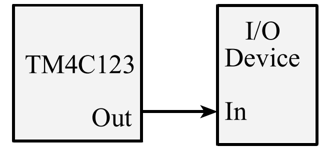
`0x0000050A 4809      LDR   r0,[pc,#36]`
`0x0000050C 6800      LDR   r0,[r0,#0x00]`
`0x0000050E F0800008  EOR   r0,r0,#0x08`
`0x00000512 4908      LDR   r1,[pc,#32]`
`0x00000514 F8C103FC  STR   r0,[r1,#0x3FC]`
`0x00000518 E7F7      B     0x0000050A`
	`int main(void){`
	`// some initialization code`
	`  while(1){`
	`    GPIO_PORTF_DATA_R ^= 0x08;`
	`  }`
	`}`

There is a critical section in this code. Assume each instruction takes exactly 2 bus cycles to execute (25ns). Also, assume there are no other interrupts. Estimate the probability that an error will occur due to the critical section for each instance of the SysTick interrupt. Hint: define the actual locations of the critical section and determine the probability of being in that section. Show your work.

(5) Question 6. Explain how the connection socket and client socket are used the **https://www.blynk.cc/** communication.

(10) Question 7. You wish to connect a device to a GPIO output of the TM4C123. These are the parameters of the I/O device:

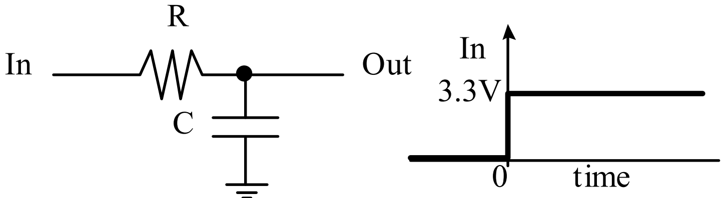
I_{IL} = 1mA, I_{IH} = 1mA,
V_{IL} = 0.3V, V_{IH} = 2.0 V



Can you directly connect a TM4C123 output to this device? Select Yes or No:

If yes, prove it. If no, show at least one parameter/equation not satisfied.

(10) Question 8. Consider the following *RC* circuit. The input, **In**, is a 3.3-V step at time 0. Give the equation for **Out** as a function of time, *R*, and *C* (for times greater than 0). This in essence is what we have on every digital signal where we connect an output pin to an input pin. *R* is the output impedance of the output pin, and *C* is the input capacitance of the input pin. Partial credit for sketching the output.



(25) Question 9. Consider the interaction between these three ISRs. You may assume the priority of these three interrupts are initially equal. Assume Port A and B are both configured as inputs with edge triggered interrupts enabled. The Port A handler is invoked on any change of Port A, and the Port B handler is invoked on any change of Port B. Each edge-triggered interrupt will input data and send it to the SysTick ISR. Assume `ProcessA` and `ProcessB` are complex functions that each take about 1000us to execute. SysTick ISR receives the data and processes the data depending on whether the data came from Port A or Port B. The average time between changes on Port A is greater than 5000us, and B is also greater than 5000us. So, there is plenty of time on average to process all the data. However, the minimum time between successive changes on a single port can be as short as 50us. Furthermore, an input from A can occur at the same time or nearly the same time as an input from B.

```
int32_t Data=0;  // semaphore
int32_t Flag=0;

void GPIOPortA_Handler(void){
  GPIO_PORTA_ICR_R = 0xFF;
  Data = GPIO_PORTA_DATA_R;
  Flag = 1;              // Send Port A data
}

void GPIOPortB_Handler(void){
  GPIO_PORTB_ICR_R = 0xFF;
  Data = GPIO_PORTB_DATA_R;
  Flag = 2;              // Send Port B data
}

void SysTick_Handler(void){
  if(Flag == 1){
    ProcessA(Data);     // Receive and process Port A data
    Flag = 0;
  }
  if(Flag == 2){
    ProcessB(Data);     // Receive and process Port B data
    Flag = 0;
  }
}
```

If the time between changes is always very large, this solution operates properly. However, if the time between successive inputs on one port temporarily drops below 1000us (even with an average time greater than 5000us), data is lost. Furthermore, if the time between a change on A and a change on B drops below 1000us, data is also lost.

Rewrite the three ISRs so no data is lost, given the timing specifications above. You may call any function(s) listed in the book or in the starter code for the class without showing the implementation(s).

You may change the priorities of the three interrupts. Specify new priorities as comments in the code.

```
void GPIOPortA_Handler(void){
  GPIO_PORTA_ICR_R = 0xFF;
```

```
void GPIOPortB_Handler(void){
  GPIO_PORTB_ICR_R = 0xFF;
```

```
void SysTick_Handler(void){
```

(20) Question 10. There are 8 digital inputs connected to Port B (assume the inputs do not bounce). If any of the inputs has a rising edge you will call a user-supplied function. Similarly, if any of the inputs has a falling edge you will call a different user-supplied function. You may assume rising edges on one input do not occur at the same time (or nearly at the same time) as falling edges on other input. You are asked to configure all 8 bits of Port B for both rising and falling edge interrupts. You can neglect priority in this question. You need not be friendly because you use all of Port B.

The prototype for your function is

```
void Init((void)*rise(void),(void)*fall(void));
```

The following code illustrates how the main program uses your system.

```
void myRising(void){ // you do not write this
  // stuff
}
void myFalling(void){ // you do not write this
  // other stuff
}
int main(void){ // you do not change the main program
  Init(&myRising,& myFalling);
  EnableInterrupts();
  while(1){
  }
}
```

Put your answers in the boxes on the next page.

(10) Part a) Complete the initialization routine. You may add globals.

```
void Init((void)*rise(void),(void)*fall(void)){
  SYSCTL_RCGCGPIO_R |= 0x02;
  delay = SYSCTL_RCGCGPIO_R;
  GPIO_PORTB_PCTL_R = 0x00000000;
  GPIO_PORTB_DIR_R  = 0x00;
  GPIO_PORTB_DEN_R  = 0xFF;
  GPIO_PORTB_IS_R  =       ;
  GPIO_PORTB_IBE_R =       ;
  GPIO_PORTB_IEV_R =       ;
  GPIO_PORTB_IM_R =        ;
  NVIC_EN0_R =             ;
```

```
// Put globals here
```

(10) Part b) Show the interrupt service routine

```
void GPIOPortB_Handler(void){
```