Jonathan W. Valvano First:_____ Last:_____
November 18, 2011, 2:00pm-2:50pm. Open book, open notes, calculator (no laptops, phones, devices with screens larger than a TI-89 calculator, devices with wireless communication). You have 50 minutes, so please allocate your time accordingly. ***Please read the entire quiz before starting***.

(5) Question 1. The starter file for Lab 6 has a 4.7 μF capacitor from 3.3V to ground. What is the purpose of this capacitor? All choices are true statements; pick the one that best answers the question.

A) The regulator needs capacitance at its output in order to stabilize the +3.3V supply. The manufacturer of each regulator will suggest appropriate capacitance values and capacitor types at its input and output.

B) A pi filter (CLC) can be used to separate digital noise from analog circuits. It works by preventing high frequency current spikes on the digital circuits from causing voltage spikes on the analog circuits. Similar filters can be made with ferrite bead cores.

C) CMOS logic needs charge whenever a digital line rises or falls. Current through the resistance of the wire will cause a voltage drop. The closer the capacitor is to the chip, the less voltage drop there will be on the V_{dd} pin.

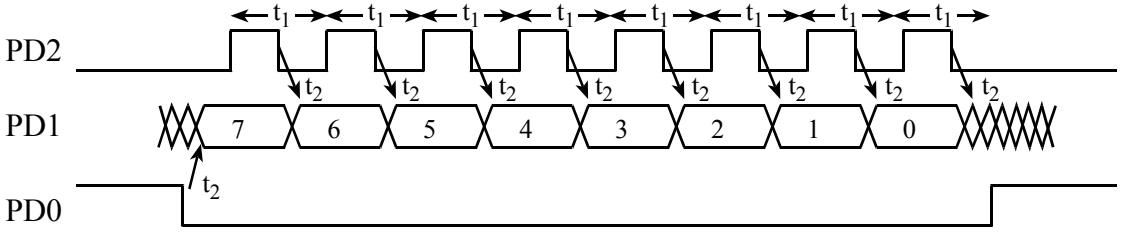
D) Capacitors are extremely important for the phase lock loop (PLL). In particular, the proper choice of capacitors is required for the PLL to lock. In fact, one usually knows in advance what frequency the PLL will be used and tunes the capacitors for that frequency.

E) Capacitors are extremely important for the crystal oscillator (OSC1 and OSC0). In particular, the proper choice of capacitors is required for the bus clock to oscillate. The position of the capacitors in your layout is critical for the clock to operate.

F) Capacitors are an important part of a low pass filter. For a single pole low pass filter made with a resistor and capacitor, we can estimate the time constant as R*C; the frequency response of such a system is about $1/(2\pi RC)$.

Put your answer in the box.

(40) Question 2. The goal is to receive synchronous serial data using edge-triggered interrupts. The Stellaris LM3S is a slave input device meaning all three signals PD2, PD1, and PD0 are inputs to the microcontroller. The time t_1 is [10, 1000 μs]. The time t_2 is [10, 200 ns]. The PD2 clock has a 50% duty cycle. The PD2-PD0 pins do not constitute an SSI port; you will solve this interface with regular input ports and edge-triggered interrupts. The 8-bit data follows this protocol. The bus frequency is 50 MHz.



You may use these bit-banded definitions

```
#define PD0   (*((volatile unsigned long *)0x40007004))
#define PD1   (*((volatile unsigned long *)0x40007008))
#define PD2   (*((volatile unsigned long *)0x40007010))
```

(5) Part a) Use the macro in Program 3.9 to create a FIFO to pass data from background to foreground. Allocate 64 elements for the FIFO. When the main program wishes to consume synchronous serial data it will execute this function. This function returns one byte of data, not one bit.

```
unsigned char SS_InChar(void){ unsigned char data;
  while(SSFifo_Get(&data) == 0){};  // FIFO empty if returns 0
  return(data);
}
```

(20) Part b) Show the initialization code that configures PD2, PD1, and PD0. In particular fill in C code for the 10 boxes. It should arm and enable the appropriate edge-triggered interrupt. Make this ISR the highest possible interrupt priority. You may add private global variables. Be friendly. No backward jumps are allowed in the ISR.

```
void SS_Init(void){
  SYSCTL_RCGC2_R
  SSFifo_Init();
  GPIO_PORTD_DIR_R
  GPIO_PORTD_DEN_R
  GPIO_PORTD_IS_R
  GPIO_PORTD_IBE_R
  GPIO_PORTD_IEV_R
  GPIO_PORTD_IM_R
  NVIC_PRI0_R
  NVIC_EN0_R
// initialization of variables
```

```
  EnableInterrupts(); }
```

(15) Part c) Show the edge-triggered interrupt service routine. Ignore input data if the PD0 line is high. This is the highest priority ISR in the system. No backward jumps are allowed in the ISR. You do not need to show the main program.

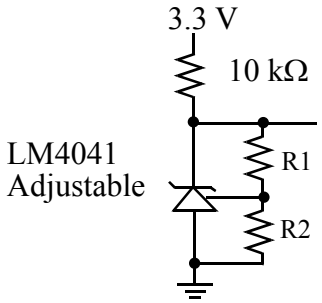
```
void GPIOPortD_Handler(void){
```

(10) Question 3. The output of a linear transducer is V_1, and V_1 is connected as the input to an amplifier. The amplifier has a transfer function of $V_2 = 100 (V_1+0.02)$. The range of V_1 is -0.02 to 0.02 V. V_2 is connected as the input to an ADC. The ADC range is 0 to 4 V, and ADC precision is 10 bits. The sampling rate is 1000 Hz. What is the maximum allowable voltage noise of the amplifier, referred to its input? Show your work.

(5) Question 4. What is the best definition for the ADC sequencer?
A) When the successive approximation converter is operating, the sequencer performs the bit sequence from most significant to least significant bit conversion.
B) When the ADC samples multiple channels the sequencer specifies whether it samples from low to high, or from high to low. E.g., sample channels 0, 1, 2, 3 or 3, 2, 1, 0.
C) When the ADC samples multiple channels, the sequencer determines which channels will be sampled.
D) The sequencer specifies which of the 8 trigger sources will be used to sequence the ADC.
Put your answer in the box.

(5) Question 5. What is the best description for of a flash ADC converter?
A) The ADC uses a low precision ADC, a low-precision DAC, and oversampling so that the DAC output equals the analog input in a time-averaged sense.
B) An n-bit ADC uses 2^n analog comparators running in parallel so all ADC bits are determined at the same time.
C) The ADC speed is linear with the number of bits.
D) It is the type of converter used in the Stellaris LM3S microcontroller. It can sample up to a million samples per second.

(20) Question 6. Design an analog circuit with the following transfer function $V_{out} = 100*(V_{in}+0.02)$. The input is a single voltage (not differential). The input range is -0.02 to 0.02 V and the output range is 0 to 4V. Use an analog reference and one rail to rail op amp (not an instrumentation amp). Show your work and label all chip numbers and resistor values, including R1 and R2. You do not have to show pin numbers.



(15) Question 7. The Timer1 ISR executes every 1 ms. A 12-bit DAC is interfaced to an SSI port. The range of the DAC is 0 to 3 V (0 → 0V, 4095 → 3V). The SSI is configured in master mode; the SSI clock is 1 MHz. The SSI data width is 16 bits. I.e., each SSI output is 16 bits long and produces one DAC output. The **Buffer** has a 16-point triangle wave. Assume PD0 is initialized as an output pin. During initialization, which you do not have to write, the DAC output is set to 2048 (1.5V).

```
#define PD0   (*((volatile unsigned long *)0x40007004))
const unsigned short Buffer[16] = {  // difference between points is 512
  2560, 3072, 3584, 4095, 3584, 3072, 2560, 2048,
  1536, 1024,  512,    0,  512, 1024, 1536, 2048
};
void Timer1A_Handler(void){
  PD0 = 1;
  TIMER1_ICR_R = TIMER_ICR_TATOCINT;// acknowledge timer0A timeout
  for(i=0;i<16;i++) DAC_Out(Buffer[i]);
  PD0 = 0;
}
```

Assume the first interrupt is triggered at time equals 0. Draw the output of the DAC and the PD0 output as a function of time for at least two interrupts. Be as accurate as you can on the time axis. E.g., clearly specify how long each interrupt takes.