Jonathan W. Valvano First:__________________ Last:____Solution________

November 18, 2016, 10:00-10:50am. 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***.

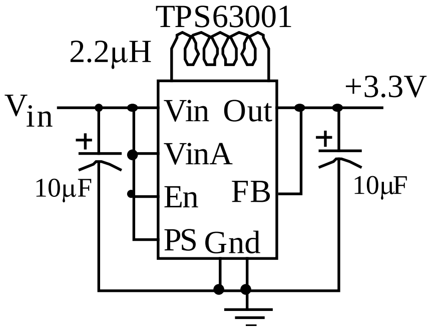
(10) Question 1. Design a circuit that takes a 1.5V alkaline battery and produces a 3.3V power supply for an embedded system. In sleep mode, the system needs 0.05 mA of current. In active mode, the system requires 100 mA of current. Show all part numbers, but do not worry about exact values for the resistors, capacitors and inductors. It runs in sleep mode 99% of the time and runs in active mode 1% of the time.

See Table 9.5 in the book

Need output voltage 3.3V (linear regulator would need input >3.3V)

Need output current more than 200 mA

Need boost converter, because output larger than input



(10) Question 2. We have a temperature data acquisition system like Lab 9. Let *n* be the 12-bit ADC result, and assume the system is linear. The measured temperature calculated with a linear equation **m****n*+**b**, where **m** and **b** are calibration coefficients. Consider these factors that might affect performance.

A) Switching from the 12-bit ADC on the TM4C123 to the 14-bit ADC on the MSP432
B) Varying the TM4C123 bus frequency from 40 MHz to 80 MHz using the PLL
C) Electromagnetic field noise pickup
D) Varying the ADC sampling rate from 100 Hz to 10 kHz
E) Calibration drift, meaning slope **m** and offset **b** drift over time.
F) Thermal noise in the analog electronics
G) Using floating point versus fixed point in the linear equation **m****n*+**b**

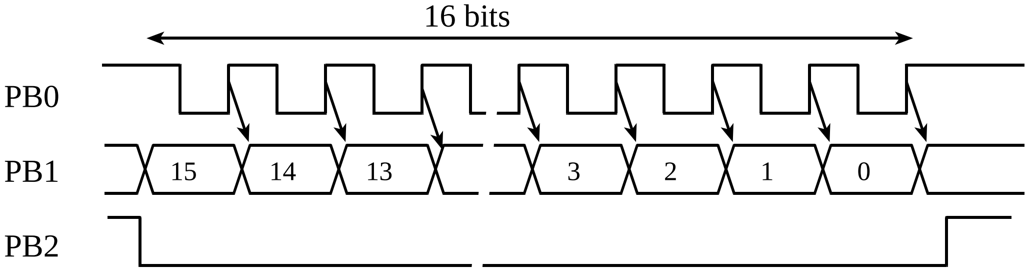
(5) Part a) Which of the above factors affect **accuracy**? Choose all factors that affect temperature measurement accuracy. Place the appropriate letters in the box.

A, C, E, F

(5) Part b) Which of the above factors affect **resolution**? Choose all factors that affect temperature resolution. Place the appropriate letters in the box.

A, C, F

(20) Question 3. The goal is to receive synchronous serial data using regular input pins (not the built-in SSI module). The PB2-PB0 pins do not constitute a regular SSI port; you will solve this interface with three GPIO input ports. The 16-bit data follows this protocol. All three signals are outputs from the external device and inputs to your TM4C123.



You may use any of the standard definitions in the **tm4c123gh6pm.h** file. In addition, you may use these bit-specific definitions

```
#define PB0   (*((volatile uint32_t *)0x40005004))
#define PB1   (*((volatile uint32_t *)0x40005008))
#define PB2   (*((volatile uint32_t *)0x40005010))
```

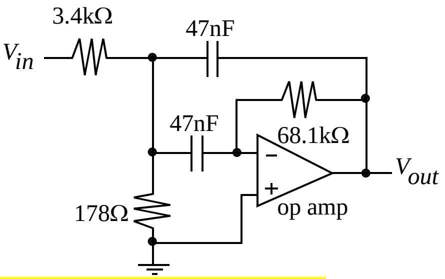
Assume the GPIO Port B has already been initialized so PB2, PB1, and PB0 are simple GPIO input. Show the function that inputs one 16-bit data using the above SSI protocol. You should first wait for the chip select on PB2, and then read 16 bits from PB1 using the clock on PB0 as synchronization. You may assume the software execution is fast compared to the clock period. You may assume the device always clears PB2 low, sends exactly 16 bits, and then sets PB2 high. Use comments to explain your approach.

```
uint16_t SSI_Input(void){
uint16_t data=0;
uint32_t i;
  while(PB2 == 0x04){};   // wait for chip select
  for(i=0;i<16;i++){
    data = data<<1;       // MSB first
    while(PB0 == 0x01){}; // wait for falling edge
    data = data|(PB1>>1); // read data bit
    while(PB0 == 0x00){}; // wait for rising edge
  }
  while(PB2 == 0){};      // wait end of transmission
  return data;
}
Comments
1) All pins are input, you cannot, should not write to pins
2) You should shift before loading next bit
3) To synchronize with clock you need to wait for fall and wait for rise
```

(15) Question 4. This software samples the analog input on PD3 at the rate specified by the period parameter. This is Program 8.8 in the book. Modify this software so it samples the input on **PE0**, which is analog channel 3. Cross out parts of the code you wish to delete and insert necessary additions.

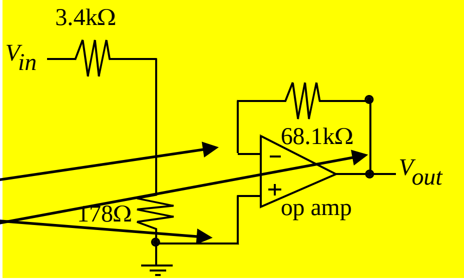
```
void ADC0_InitTimer0ATriggerSeq3~~PD3~~PE0(uint32_t period){
  volatile uint32_t delay;
  SYSCTL_RCGCADC_R |= 0x01;
  SYSCTL_RCGCGPIO_R |= ~~0x08~~0x10;   // PORTE
  delay = SYSCTL_RCGCGPIO_R;
  GPIO_PORT~~D~~E_DIR_R &= ~~~0x08~~0x01;  // PE0
  GPIO_PORT~~D~~E_AFSEL_R |= ~~0x08~~0x01; // PE0
  GPIO_PORT~~D~~E_DEN_R &= ~~~0x08~~0x01;  // PE0
  GPIO_PORT~~D~~E_AMSEL_R |= ~~0x08~~0x01; // PE0
  ADC0_PC_R = 0x01;
  ADC0_SSPRI_R = 0x3210;
  SYSCTL_RCGCTIMER_R |= 0x01;
  delay = SYSCTL_RCGCGPIO_R;
  TIMER0_CTL_R = 0x00000000;
  TIMER0_CTL_R |= 0x00000020;
  TIMER0_CFG_R = 0;
  TIMER0_TAMR_R = 0x00000002;
  TIMER0_TAPR_R = 0;
  TIMER0_TAILR_R = period-1;
  TIMER0_IMR_R = 0x00000000;
  TIMER0_CTL_R |= 0x00000001;
  ADC0_ACTSS_R &= ~0x08;
  ADC0_EMUX_R = (ADC0_EMUX_R&0xFFFF0FFF)+0x5000;
  ADC0_SSMUX3_R = ~~4~~3;       // channel 3
  ADC0_SSCTL3_R = 0x06;
  ADC0_IM_R |= 0x08;
  ADC0_ACTSS_R |= 0x08;
  NVIC_PRI4_R = (NVIC_PRI4_R&0xFFFF00FF)|0x00004000;
  NVIC_EN0_R = 1<<17;
  EnableInterrupts();
}
void ADC0Seq3_Handler(void){
  ADC0_ISC_R = 0x08;
  Fifo_Put(ADC0_SSFIFO3_R);
}
```

(10) Question 5. Consider the following circuit with two 47-nF capacitors. This is a regular op amp powered with +12 to -12 V. This circuit does pass input signals at 1 kHz with gain 10; notice that $1/(2\pi*3.4k*47nF)=1$ kHz. Assume the input is V_{in} = 1V $\cos(2\pi ft)$.



(5) Part a) What will be the approximate output of this circuit for frequencies much much smaller than 1 kHz? Show your work.

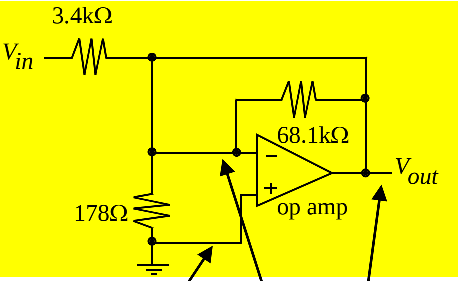
At lower frequencies the capacitors are open, and the equivalent circuit is on the left. Apply op amp rules: no current into op amp inputs, V+ approximately V-, so output is 0.



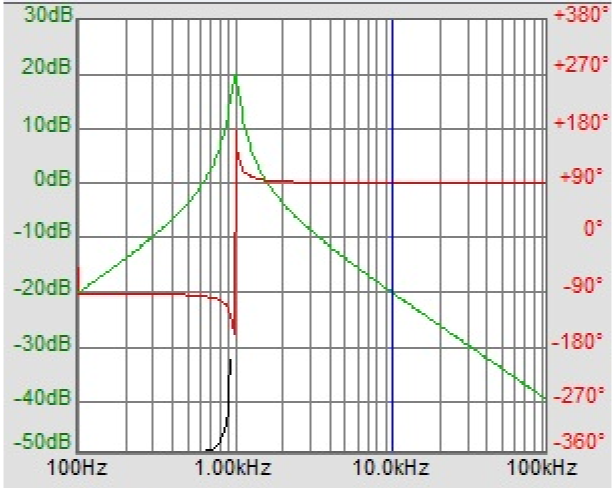
1) This is ground
2) Op amp rule (inputs equal), this will be 0V
3) Op amp rule (no current into op amp), this will be zero

(5) Part b) What will be the approximate output of this circuit for frequencies much much larger than 1 kHz? Show your work.

At higher frequencies the capacitors are shorts, and the equivalent circuit is on the left. Apply op amp rules: no current into op amp inputs, V+ approximately V-, so output is again 0.



This is a 1-kHz bandpass filter, with Q=10, gain=10.



1) This is ground
2) Op amp rule (inputs equal), this will be 0V
3) this will be zero (same as – terminal)

(15) Question 6. Assume you have a data acquisition system that measures voltage. There is a calibration procedure that populates the following static variables. Shown is an example calibration, but the system allows for dynamic calibration, so the values may change.

```
static int32_t V1=1000; // first voltage in 0.001 V units
static int32_t V2=3000; // second voltage in 0.001 V units
static int32_t N1=1274; // 12-bit ADC value corresponding to V1
static int32_t N2=3723; // 12-bit ADC value corresponding to V2
```

Write the C code that converts a 12-bit ADC measurement (0 to 4095) into the corresponding voltage in 0.001V units using a linear fit between ADC and voltage. Floating point is not allowed. The input can vary from 0 to 4095, so this is both interpolation and extrapolation. Make it as accurate as possible without causing overflow during an intermediate calculation. *Test case*: n=662 should yield a result of 500.

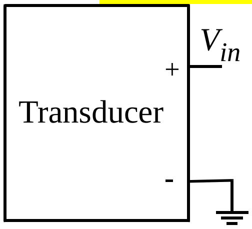
```
int32_t Convert(int32_t n){
// linear fit v = slope*n+offset
// slope = (V2-V1)/(N2-N1)  offset = V1-slope*N1
int32_t offset = V1-((V2-V1)*N1)/(N2-N1);
  return ((V2-V1)*n)/(N2-N1)+offset;
}
// this could be your exact solution from Lab 9
// interpolation will also solve extrapolation
  return V1+((V2-V1)*(n-N1))/(N2-N1);
```

(15) Question 7. Design an analog circuit that implements $V_{out} = 10*V_{in} + 1.65$V. The input voltage varies from -0.15 to +0.15V, so the output voltage will vary from 0.15 to 3.15V. The output, V_{out}, is connected to the microcontroller ADC. R1 and R2 are already chosen such that the analog reference is 1.65V. You may use any chips shown in the book or presented in class. Show your work and label all chip numbers and resistor values. You do not have to show pin numbers.

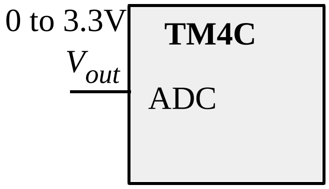
$V_{out} = 10V_{in} + 1.65$

$V_{out} = 10V_{in} + V_{ref}$

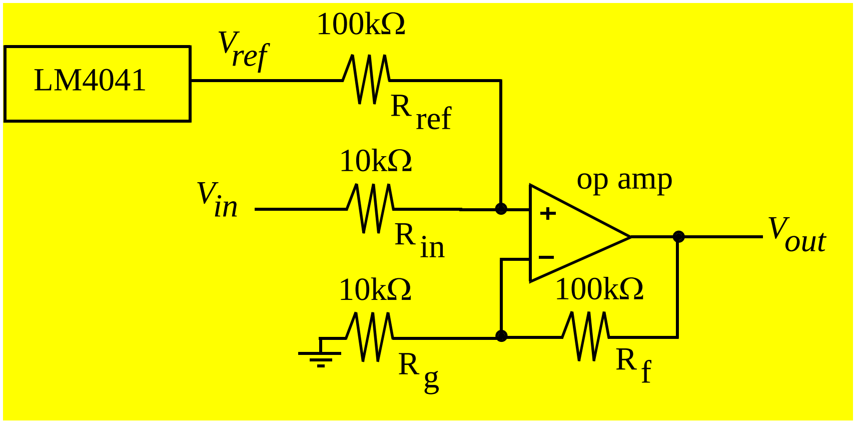
$V_{out} = 10V_{in} + V_{ref} - 10V_g$



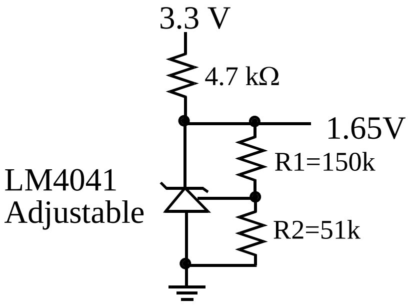
V_{in}
+
Transducer
-



0 to 3.3V
V_{out}
TM4C
ADC



LM4041
V_{ref}
100kΩ
R_{ref}
V_{in}
10kΩ
R_{in}
op amp
+
–
V_{out}
10kΩ
R_g
100kΩ
R_f



3.3 V
4.7 kΩ
1.65V
R1=150k
LM4041
Adjustable
R2=51k