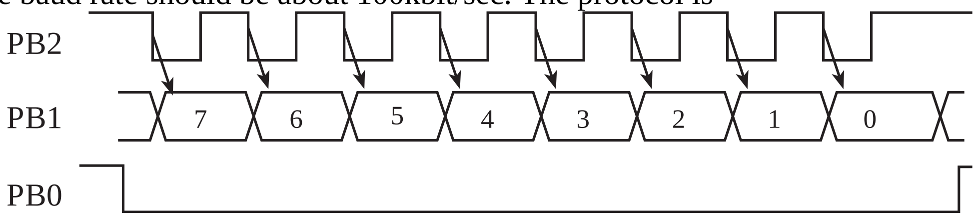
Jonathan W. Valvano First:_________________ Last:_______________

Open book, open notes. No computer, no calculator.

**(15) Question 1.** The goal is to implement an SPI-like output using just GPIO input/output (not built in SPI hardware.) The baud rate should be about 100kbit/sec. The protocol is




**Part a)** How should each of PB2 PB1 PB0 be initialized: **input** or **output**? For each output pin, specify if it should be initialized **high (H)**, **low (L)**, or **doesn't matter (X)**.

| | Input/Output? | If output, initial value: **H**, **L**, or **X**? |
|---|---|---|
| PB2 | | |
| PB1 | | |
| PB0 | | |

**Part b)** Fill in the box to specify the needed time delay for this problem. The bus clock is 80 MHz.

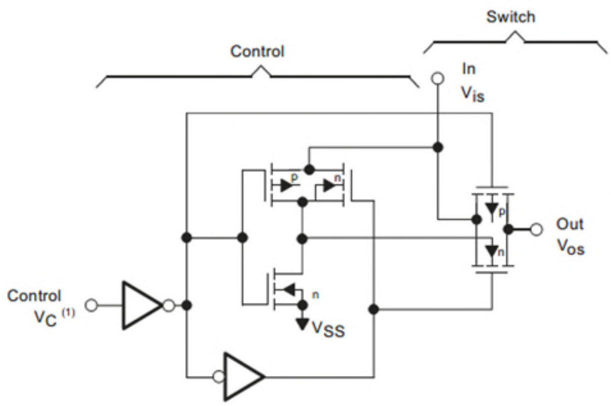
```
void Wait(void){
  NVIC_ST_RELOAD_R = [          ] ;
  NVIC_ST_CURRENT_R = 0;
  while((NVIC_ST_CTRL_R&0x00010000)==0){  }
}
```

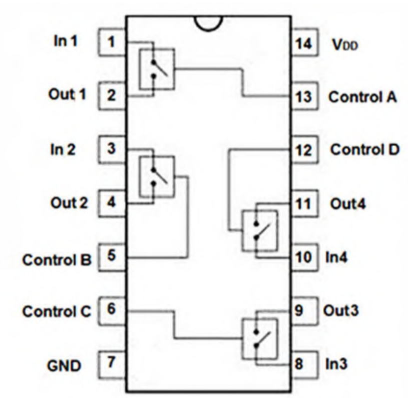
**Part c)** Assume Port B is initialized as specified in Part a) Write a function that outputs the 8-bit data using just these I/O port registers. These are bit-banded addresses where you read/write 0 or 1:

```
#define SCLK (*((volatile uint32_t *)0x420A7F88))  // PB2
#define MOSI (*((volatile uint32_t *)0x420A7F84))  // PB1
#define FSS  (*((volatile uint32_t *)0x420A7F80))  // PB0
```

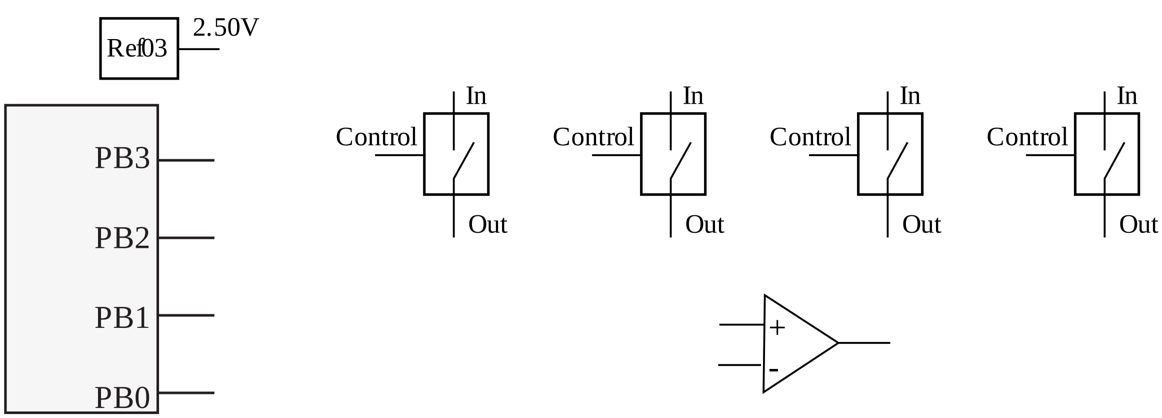
```
void Output(uint8_t data){
```

**(15) Question 2.** Build a 2-bit **thermometer coded** (**resistor string**) DAC. Rather than two GPIO output pins, you are given four GPIO output pins. You are given a REF03, some CD4066b analog switches (four are shown, you can use more less than four), op amps, resistors of any value, but no digital logic. The CD4066b **In** and **Out** pins are analog inputs or analog outputs. If the CD4066b digital **Control** signal is high the **In** pin is connected to **Out**, with a resistance of about 200 ohms. If **Control** is low, **In** and **Out** are disconnected. The REF03 is a precision voltage reference at 2.50V. The resolution should be 0.625V (2.5V/4), and the range should be 0 to 1.875V (3*0.625V).




**(10) Part a)** Show the design of the 2-bit DAC that interfaces to PB3-PB0. Specify resistor values.




**(5) Part b)** Write the C function taking a 2-bit **data** input, setting the DAC to 0, 0.625, 1.25, or 1.875V.

```
void DAC_Out(uint32_t data){
```

**(15) Question 4.** You will implement the basic idea of **input capture** on PB0 using a periodic SysTick interrupt and an edge-triggered GPIO interrupt. The period measurement resolution should be 1ms and the precision will be 64 bits. The edge-triggered interrupt occurs on each rising edge of PB0. For example, if the period of PB0 were 2 seconds, then the variable **Period** will become 2000.

**(3) Part a)** At what **rate** should the SysTick periodic interrupt run?

Frequency (Hz) =

**(3) Part b)** Define additional global variables you will need.

```
uint64_t Period; // Period of PB0 in ms
```

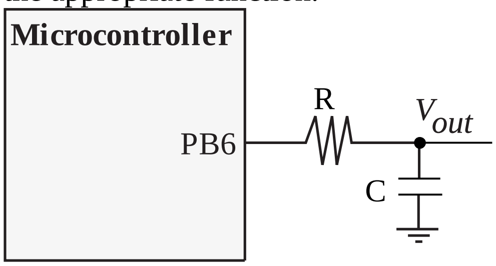
**(4) Part c)** Show the SysTick ISR code

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

**(5) Part d)** Show the edge-triggered ISR code

```
void GPIOPortB_Handler(void){
  GPIO_PORTB_ICR_R = 0x01; // acknowledge PB0 interrupt
```

**(10) Question 5.** Use this circuit to create a 16-bit DAC. R*C is 100ms.
**(5) Part a)** Using an initialization function you used in lab (also in book, also in starter codes), show the C code to initialize the 16-bit DAC. You do not have to show the function definition, just make the call the appropriate function.




**(5) Part b)** Using another function you used in lab (also in book, also in starter codes), show the C code to output a 16-bit value to the DAC. You do not have to show the function definition, just make the call the appropriate function.

```
void DAC_Out(uint32_t data){
```

**(5) Question 6.** Match each problem with the best choice. If there is more than one good solution, list all good solutions.

A) Linear regulator
B) Buck regulator
C) Boost regulator
D) Shunt diode
E) Op amp
F) Instrumentation amp
G) Crystal

Most efficient way to provide power to system ……. ______

Least noisy way to provide power to system ………. ______

Provide +5V power to system from a 3.7V battery … ______

Provide 3.3V power to system from a 5V battery ….. ______

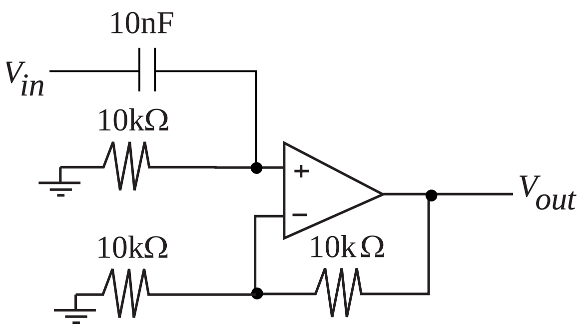
Generate a stable 16 MHz clock ……………………. ______

**(5) Question 7.** Let *X* be the desired speed, *Xstar* be the desired speed, and *U* be the actuator output. This integral controller code works but runs too slow on a processor without hardware floating point.

```
U = U + 2.25*(X-Xstar);
```

Rewrite the C code using binary fixed point.

**(5) Question 8.** Consider this analog filter. 10nF = $10^{-8}$F. 10k is $10^{3}$ ohms. R*C is 10ms. 1/10ms is 100 Hz. **Gain** is defined as $|V_{out}|/|V_{in}|$.



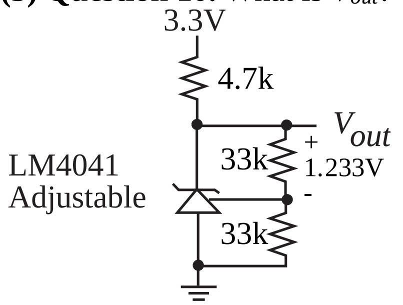

**Part a)** What is the gain at f much less than 100 Hz?

**Part b)** What is the gain at f = 100 Hz?

**Part c)** What is the gain at f much larger than 100 Hz?

**(10) Question 9.** Build an analog circuit with $V_{out} = 1000*(V_2-V_1)$. Show chip numbers and resistor values. The circuit should have a large input impedance.

**(5) Question 10.** What is $V_{out}$? You may assume the circuit is not connected to anything. Show your work.




**(5) Question 11.** The function **`Put`** is only called at one place in an interrupt service routine (producer), and the function **`Get`** is only called at one place in the main program (consumer). This FIFO can store up to 15 elements. There is a bug somewhere in this code

```
static uint32_t PutI=0;
static uint32_t GetI=0;
static int32_t FIFO[16];
```

| | |
|---|---|
| `int Put(int32_t data){` | `int Get(int32_t *datapt){` |
| `  if((PutI+1)&0x0F)==GetI) return 0;` | `  if(PutI==GetI) return 0;` |
| `  FIFO[PutI] = data;` | `  *datapt = FIFO[GetI];` |
| `  PutI = (PutI-1)&0x0F;` | `  GetI = (GetI-1)&0x0F;` |
| `  return 1;` | `  return 1;` |
| `}` | `}` |

Find the bug in this code and fix it by **changing just one line**