Jonathan W. Valvano First Name: _______________ Last Name:_____________________
May 12, 1999, 9am-12noon

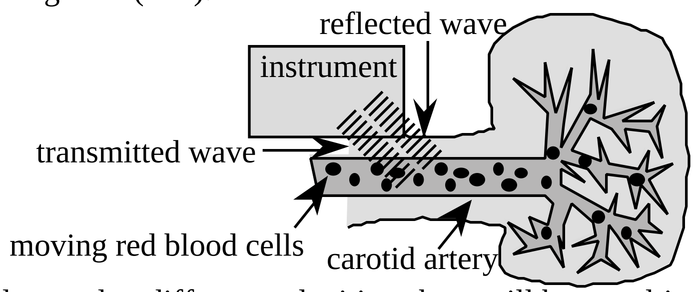
This is an open book, open notes exam. You may put answers on the backs of the pages, but please don't turn in any extra sheets. You have 3 hours, so please allocate your time accordingly.

(10) Question 1. Design an analog circuit with the following specifications

transfer function	$V_{out} = 25 \bullet (V_1 - V_2)$
constrained input	$V_1 \geq V_2$ and $(V_1 - V_2) \leq 0.2V$
constrained output	$0 \leq V_{out} \leq +5V$
large input impedance	$Z_{in} \geq 1M\Omega$

The circuit will be used in a data acquisition system with a sampling rate, f_s, of 10 Hz. Only the +5 V supply can be used (no ±12V supplies). The entire circuit should be built with a single TLC2274 chip (four op amps). Give chip numbers but not pin numbers. Specify the type and tolerance of resistors and capacitors. Show the analog circuit.

(30) Question 2. Design a noninvasive doppler ultrasound blood velocity meter. You will send a 4MHz ultrasonic wave into the body aimed at a major artery like the carotid artery in the neck as shown in the figure. When the sound reflects off a moving red blood cell there will be a frequency shift. The frequency difference between the transmitted and reflected wave is $\Delta \mathbf{f} = 2 \cdot \mathbf{v} \cdot \mathbf{f} \cdot \cos(\theta)/\mathbf{c}$. **v** is the velocity of the red blood cells (in cm/s). **f** is the frequency of the transmitted ultrasound (4MHz). **c** is the speed of sound in the biologic tissue (150,000 cm/s). θ is the angle between the sound beam and the moving cells (45 ˚).



Since the red blood cells travel at different velocities, there will be a multitude of frequency shifts. Luckily the analog electronics will select the largest doppler shift, which represents the fastest moving blood cell. Consequently the instrument will measure peak velocity. The measurement range is 0 to 200 cm/sec, resulting in a range of doppler frequencies of 0 to 7542 Hz. The desired velocity measurement resolution is 1 cm/sec. The analog circuit provides for us a sine wave with an amplitude of ±1 V at the doppler frequency (e.g., $0 \le \Delta \mathbf{f} \le 7542$ Hz.) Measure the velocity over and over and store the measurement result in a global variable called v.

```
unsigned short v;   // measured velocity in cm/sec
```

(10) Part a) Show the electronic interface between the ±1 V sine wave (labeled **Δf**) and the input capture pin PT0. You may use a ±12V supply. Give chip numbers but not pin numbers.

(10) Part b) Show the ritual that initializes the system. You must use frequency measurement. One of the important issues is to configure the frequency measurement to match the desired velocity resolution of 1 cm/sec. Add global variables as needed.

(10) Part c) Show the input capture and output compare handlers that implement the velocity meter. Set the global variable `v`, and call the following function after each measurement. You do not need to show the implementation of `LCDout()`.

```
LCDout("v=",v," cm/sec");
```

(25) Question 3. Design a digital circuit that implements an I/O port. This port will attach directly to the address/data bus of a MC68HC812A4 running at 8 MHz in expanded narrow mode. There will be two 8-bit registers, implemented with two chips like the 74HC374. One 8-bit register is the direction register and the other 8-bit register is the data register. You may use any 74HC digital logic chip. You will use the built-in chip select CS2 to place the device in the $0380 to $03FF address range. Some of the features of your I/O port include

- 8 pins
- 8 individual data direction bits, one for each pin
- if direction register bit is 0, then readable input at address $0380
- if direction register bit is 1, then readable output at address $0380
- readable direction register at address $0381
- if you write first, then change the direction to output, then the data output becomes valid

(20) Part a) Show the digital logic interface. Give chip numbers, but not pin numbers.

(5) Part b) Give the ritual program to initialize CS2 with the appropriate number of clock stretches.

(20) Question 4. In this problem we start with the interrupt-driven DAS from Section 11.9. The C statement `TC5=TC5+Rate;` guarantees that OC5 interrupts are requested at a regular rate. But what really matters is the time at which the A/D converter is started. In particular we need to start the A/D at a regular rate. The goal is to experimentally measure the *minimum*, and *maximum* intervals between the start of one A/D measurement and the start of the next A/D sample in µsec. Unfortunately any software that runs with interrupts disabled will postpone an output compare interrupt service causing a time shift in the data. We expect the time delay between calls to `Adin()` to be always less than 32 ms. In particular you will measure the time between calling `Adin()` during one interrupt and calling it again during the next interrupt.

```
// Program from Section 11.9
#define Rate 2000
#define OC5  0x20
unsigned int x[4];  // MACQ (mV)
unsigned int d;     // derivative (V/s)

#pragma interrupt_handler TOC5handler()
void TOC5handler(void){
   TC5=TC5+Rate;   // Executed every 1 ms
   TFLG1=0x20;     // ack OC5F
   x[3]=x[2];  // shift MACQ data
   x[2]=x[1];  // units of mV
   x[1]=x[0];
   x[0]=Adin(); // current data
   d=x[0]+3*x[1]-3*x[2]-x[3];} // mV/ms

void ritual(void) {
asm(" sei");     // make atomic
  TIOS|=OC5;     // enable OC5
  TSCR|=0x80;    // enable
  TMSK2=0x32;    // 500 ns clock
  TMSK1|=OC5;    // Arm output compare 5
  TFLG1=OC5;     // Initially clear C5F
  TC5=TCNT+Rate; // First one in 1 ms
asm(" cli"); } }
```

```
// With your debugging instruments
#define Rate 2000
#define OC5  0x20
unsigned int x[4];  // MACQ (mV)
unsigned int d;     // derivative (V/s)
#pragma interrupt_handler TOC5handler()
void TOC5handler(void){
   TC5=TC5+Rate;   // Executed every 1 ms
   TFLG1=0x20;     // ack OC5F
   x[3]=x[2];  // shift MACQ data
   x[2]=x[1];  // units of mV
   x[1]=x[0];
   DebugMeas();
   x[0]=Adin(); // current data
   d=x[0]+3*x[1]-3*x[2]-x[3];} // mV/ms
void ritual(void) {
asm(" sei");     // make atomic
   DebugInit();
  TIOS|=OC5;     // enable OC5
  TSCR|=0x80;    // enable
  TMSK2=0x32;    // 500 ns clock
  TMSK1|=OC5;    // Arm output compare 5
  TFLG1=OC5;     // Initially clear C5F
  TC5=TCNT+Rate; // First one in 1 ms
asm(" cli"); } }
```

(5) Part a) In addition to the following two *public* global variables, please specify any additional *privite* global variables you will require.

```
unsigned short Minimum,Maximum; // execution times in usec
```

(15) Part b) Write the two debugging instruments. `DebugInit();` will initialize the globals. The timer is already initialized to count at 2 MHz. `DebugMeas();` will perform a measurement and update the two global variables: `Minimum,Maximum`.

(*bonus*) Part c) Under what condition might the minimum time be much less than expected?

(15) Question 5. In this problem you will design a full-duplex serial channel between two microcomputers. The goal is to allow full-duplex communication without connecting the two grounds. The distance between the two computers is short enough so that two 4N45 optocouplers can be used.

(10) Part a) Show the interface between the two microcomputers. Label chip numbers, but not pin numbers. Notice the two separate and distinct ground symbols. Carefully label the ground connection on any interface chips you use.

(5) Part b) What is the maximum baud rate possible. Explain your answer