

Jonathan W. Valvano First: _____ Last: _____
 November 5, 1999, 11:00am-12:45pm

This is an open book, open notes exam. You must put your answers on these pages only, you can use the back. You have 90 minutes, so please allocate your time accordingly.

Please read the entire quiz before starting.

(40) Question 1. In this problem you will design an ultrasonic doppler velocity meter. The goal is to measure velocity, $0 \leq v \leq 100$ cm/sec. The desired velocity resolution, Δv , is 1 cm/sec. The doppler transducer produces a ± 5 V sine wave. The frequency, f , is linearly related to velocity, v :

$$f \text{ (in Hz)} = 400 \cdot v \text{ (in cm/sec)}$$

Therefore, the frequency range is 0 to 40 kHz. The resulting frequency measurement resolution, Δf , is $400 \cdot \Delta v$ or 400 Hz. You will build this system using the input capture system and **period measurement**. You will use input capture channel PT2, to measure period. You will use output compare channel PT3 to handle the special case when the velocity is less than 1 cm/sec. The two interrupt service routines (PT2 and PT3) will calculate velocity and continuously update the following global variable with the current velocity measurement.

unsigned char v; // velocity, range is 0 to 100 and resolution is 1 cm/sec

The following table shows the corresponding frequency and period values:

v (cm/sec)	f (Hz)	p (μ s)
0	0	infinite
1	400	2500
2	800	1250
50	20000	50
99	39600	25.25
100	40000	25

(5) Part a) What period measurement resolution do you need to distinguish 99 from 100 cm/sec?

(10) Part b) Show the hardware interface between the ± 5 V sine wave and the Port T PT2 input capture digital input. Make the hysteresis larger than 10 mV. Show chip numbers, power supply connections, resistor values, but not pin numbers.

(10) Part c) Write the ritual subroutine that initializes the interface. Please define and initialize any additional data structures that you require. Don't worry about the first measurement.

(15) Part d) Write the interrupt service routines that measure velocity. Even though you probably need to write this in assembly to actually make it work, please write the ISRs in C. Don't worry about the interrupt vectors, just use names like `T0C3handler()` and `TI C2handler()`.

(20) Question 2. In this problem you will write a C function that implements this digital filter.

$$y(n) = 0.950625x(n) - 1.90125x(n-1) + 0.950625x(n-2) + 1.9y(n-1) - 0.9025y(n-2)$$

You may assume your function is called at the appropriate sampling frequency, with the appropriate new data value. You may assume each new data point is a 12-bit unsigned value and the filter gain is less than 1 (so the output of the filter will always be less than 4095.) The function prototype is

unsigned short filter(unsigned short data);

// input parameter is the new x(n)

// output parameter is the resulting calculation of y(n)

You will implement the MACQ as static local variables holding $x(n-1)$, $x(n-2)$, $y(n-1)$, $y(n-2)$

(10) Part a) Re-write the digital filter using fixed-point approximations.

(10) Part b) Show the C code for the filter function. No floating point is allowed. Do not include output compare ISR or ADC initialization, just the digital filter implementation.

(40) Question 3. In this problem you will interface a 74LS374 as an output port to the MC68HC812A4. In other ports, when software writes to the port address of \$3000, the 8-bit data value is copied into the 8 Q outputs of the 74LS374. For example, the following assembly code will set the 74LS374 bits 7-4 high and bits 3-0 low:

```
ldaa #$F0
staa $3000
```

The 6812 is running at 8 MHz. The 74LS374 output enable, OE, will be grounded so that its Q outputs will be continuously available. Your interface will be "write-only", participating in write cycles, but not read cycles. The 74LS374 timing can be found in Section 9.3 (figure 9.21).

(5) Part a) What 6812 operating mode will you use (e.g., single chip, expanded narrow, expanded wide, or special peripheral)?

(10) Part b) Design the minimal cost positive logic address decoder for your interface assuming the following modules. You will **not** use the built-in address decoder of the 6812. In this part, simply show the digital equation. In the next part, you will build the decode using digital logic gates.

```
Ports  $0000 to $01FF
RAM     $0800 to $0BFF
Your    $3000 to $3000
ROM     $5000 to $FFFF
```

```
0000,000X,XXXX,XXXX
0000,10XX,XXXX,XXXX
0011,0000,0000,0000
0101,XXXX,XXXX,XXXX
011X,XXXX,XXXX,XXXX
1XXX,XXXX,XXXX,XXXX
A15,A14,A13 (or A15 A14,A12)
```

Select = $\overline{A15} \cdot \overline{A14} \cdot A13$

(5) Part c) Choose which of the following shapes will make write data available overlap write data required (See Table 9.8 and Figure 9.28 in Section 9.5.1):

- synchronized positive logic
- unsynchronized positive logic
- synchronized negative logic
- unsynchronized negative logic

(5) Part d) Show the digital interface between the 6812 and the 74LS374 including the address decoder designed in part b)

(5) Part e) How many cycle stretches are needed for this interface? Assume a 10 ns gate delay through any digital logic gate.

(10) Part f) Show the combined **write cycle** timing diagram. Show **E**, **R/W**, **A15-A0**, **Select**, **Clk**, **WDA**, and **WDR**. All signals are outputs except Write Data Required. Use arrows to signify causal relations. Show the timing delays as arrows with numbers in nanoseconds. Calculate the actual WDA and WDR intervals.