Mechatronics 3TA4

DAY CLASS — Dr. Mark Lawford
DURATION OF EXAMINATION: 1 Hour 50 min
McMaster University Midterm Examination — October 23, 2007

THIS EXAMINATION PAPER INCLUDES 4 PAGES AND 5 QUESTIONS. YOU ARE RESPONSIBLE FOR ENSURING THAT YOUR COPY OF THE PAPER IS COMPLETE. BRING ANY DISCREPANCY TO THE ATTENTION OF YOUR INVIGILATOR.

Special Instructions: The use of course notes, textbooks and any other written material is permitted during this exam. You may use the McMaster Standard Calculator. Answer all questions in the provided answer booklets. Fill in your name and student number on each booklet you use.

1. **Embedded Systems Considerations** (20 marks)

   a) (5 marks) You have a design for an embedded system that uses a MEGA234P which you can get for \$4.92/chip. If you put in another week of design time (35 person hours), you could squeeze the software into an ATtiny25 which you can get for \$2.19/chip. Assume an hour of engineering time costs the company \$40.

      (i) How many units do you have to ship to make the effort worthwhile?

      (ii) Are there any other issues you should considerations before you make your decision about switching microcontrollers?

   b) (1 mark) Do AVR microcontrollers have Harvard or Princeton Architecture?

   c) (2 marks) What are some benefits of the architecture used in the AVR?

   d) (2 marks) What are some limitations of the architecture used in the AVR?

   e) (5 marks) Explain why you have to be careful about using recursion in an embedded microcontroller like the AVR MEGA324P.

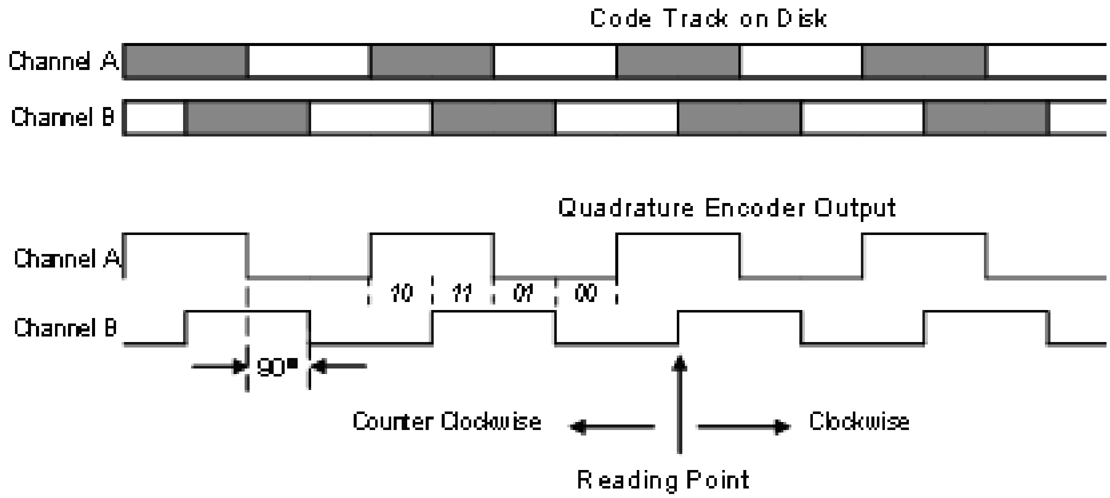
   f) (3 marks) Name 3 different peripherals commonly found built into a micro controller and for each peripheral explain in a sentence or two what it does.

   g) (2 mark) What is a benefit of having in your system a single purpose peripheral to handle a particular function? What is a drawback?

Continued on page 2

2. **Encoders** (20 marks)

Suppose you have a 24 pulse/revolution incremental encoder i.e. each channel has 24 cycles in a revolution. It has `A` and `B` channels resulting from the following pattern with the shown offsets.



a) (1 mark) What is the hexadecimal equivalent of 24 (in decimal)?

b) (1 mark) What is the decimal equivalent of 0b10111?

c) (2 marks) Assume that the encoder will only be turned in a clockwise fashion and you are going to use a counter to keep track of the position. You can assume that the counter has an asynchronous reset input. How many bits does the counter need? State any assumptions you make.

d) (2 marks) What is the resolution in degrees/pulse of the encoder?

e) (7 marks) Design the combinatorial logic for the counter's reset input that will cause the counter to reset to 0 only when the encoder has been turned 1 full revolution.

f) (7 marks) Now assume that the encoder will be rotated both clockwise and counter clockwise. How could you use an up/down counter to keep track of the position of the encoder?

3. **Reaction Timer** (20 marks)

Consider the following informal requirements for a simple reaction timer:

(i) Upon a reset signal, flash the LEDs on the development board at a few Hz and wait for a button 0 press.

(ii) At the button press, turn the LEDs off for about 2 seconds. Figure out a way to make the waiting time random, so that it is harder to predict.

(iii) After the variable time delay, turn on the LEDs, start a timer and wait for a button press. (your code should detect the cheating condition of a pressed button at t=0 and respond by returning to the flashing LEDs).

(iv) At the button press, compute the time between LED illumination and the button press, display the number of millseconds in decimal on the LCD, then waits. Displayed time should be accurate to 1 mSec.

Continued on page 3

(v) Returns to the flashing LEDs (3i above) when button 7 is pushed.

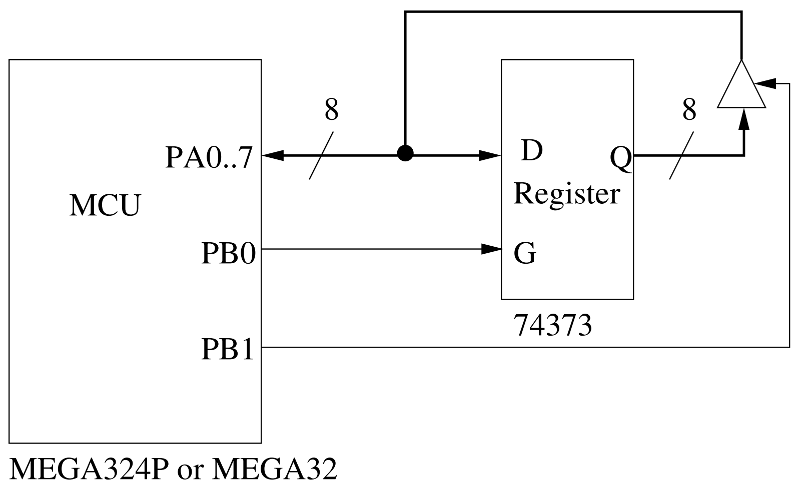
(vi) Stores the fastest time in eeprom and displays it on the LCD. Since eeprom write cycles are limited, only write to the eeprom when there is a new fastest time.

a) (15 marks) Draw a Finite State Machine that you could implement to meet the requirements. [Note: This is really a mode transition diagram.]

b) (5 marks) Why is it generally a bad idea to use software delays to implement the timing functionality of an embedded system?

4. **AVRs & Interfacing** (20 marks)

Consider the simple hardware shown below. You can choose to use either the MEGA324P or MEGA32 as the MCU:



The triangular shaped part is an 8 line tristate buffer. When `PB1` is low, the 8 lines coming in from the bottom are effectively disconnected from those at the top (8 open switches). When `PB1` is high, it effectively acts as a direct connection (8 closed switches).

The 74373 chip is just an 8 bit register. When the `G` input is high, it stores the input at `D`. Whatever value is stored then appears at the chips output `Q`.

Your goal is to write AVR C code that will write a value in an integer variable `x` in the MEGA324P (or MEGA32 - your choice!) to the register (the 74373), wait for 10ms and then read the value back into an integer variable `y`.

(a) (5 marks) What should the initial values of the Data Directions Registers be set to for PORTA and PORTB?

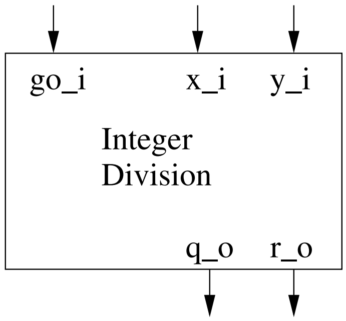
(b) (5 marks) If the MEGA324P (or MEGA32) is clocked by an 8MHz crystal, how would you set up Timer0 to get a 1 ms delay?

(c) (10 marks) Assume that you correctly configured Timer0 to generate an interrupt every 1ms and there is an interrupt service routine that sets a global boolean variable to true when this interrupt occurs. Write a *main()* function to implement the behaviour outlined above.

Continued on page 4

5. **Custom Single Purpose Processor Design** (20 marks)

Consider the following blackbox (left) and desired functionality (right) for a simple custom single purpose processor to perform integer division of `x_i` and `y_i` and return the quotient `q_o` and remainder `r_o`:




```
int i=0;
int x, y;
while (1) {
  while (!go_i);
  x = x_i;
  y = y_i;
  while (x >= y ) {
    x = x-y;
    i++; }
  q_o = i;
  r_o = x; }
```

a) (10 marks) Draw the state diagram for the processor, optimizing where possible for instructions that can be executed independently.

b) (9 marks) Separate the state diagram into a controller and datapath.

c) (1 marks) How many bits would be required to encode the state of your resulting controller? (NOTE: You do not have to compute the combinatorial logic to answer this question!).

The End