ECE331 Microcomputers (KEH) November 10, 2009
Test #2 – 100 Points Takehome Test
Due by 5 PM, Monday November 16, 2009, Under my office door
Open Textbook, Homework, Quizzes, Labs, FreeScale .PDF Manuals, and Course Notes
Dept. of Electrical and Computer Engineering
Rose-Hulman Institute of Technology

Name: ____ CM Box:_________

Required Signature: My signature below certifies that I have not received help from any person. I have not given help to any person on this exam in any shape, form, or fashion. I have not used any resources except for those listed above.

Signature:____ Date: ____________

1) **(20 Points, 1 point per blank)**
Interrupt-driven White Noise Generator Implemented in C
The hardwired circuit shown below in Figure P1 is a 31-bit right-shift register consisting of D flip-flops FF0 – FF30, whose input is formed by EXCLUSIVE OR-ing the outputs of FF2 and FF30. This sequence generator produces a "maximum length" pseudorandom binary sequence (PRBS) that will not repeat until 2^{31}-1 = 2,147,483,647 (that is over 2 Billion!) clock pulses have elapsed. The system output may be taken from the output of any flip-flop in the shift register. The (normally closed) PRESET pushbutton is used to start the shift register in the state of all 1's, since the state of all 0's is the one state that is *not* allowed in a maximal length pseudorandom binary sequence generator (since it locks the generator into a sequence that is all 0's), and so we must not let this circuit start in the all 0's state. When clocked at 20 kHz, the sequence will take (2^{31}-1)/20000/60/60 = 29.8 hours to repeat itself! Thus the binary output is a rather random sequence of 0's and 1's! If this circuit drives a loudspeaker, it will produce white noise that might be used as a sleep aid.

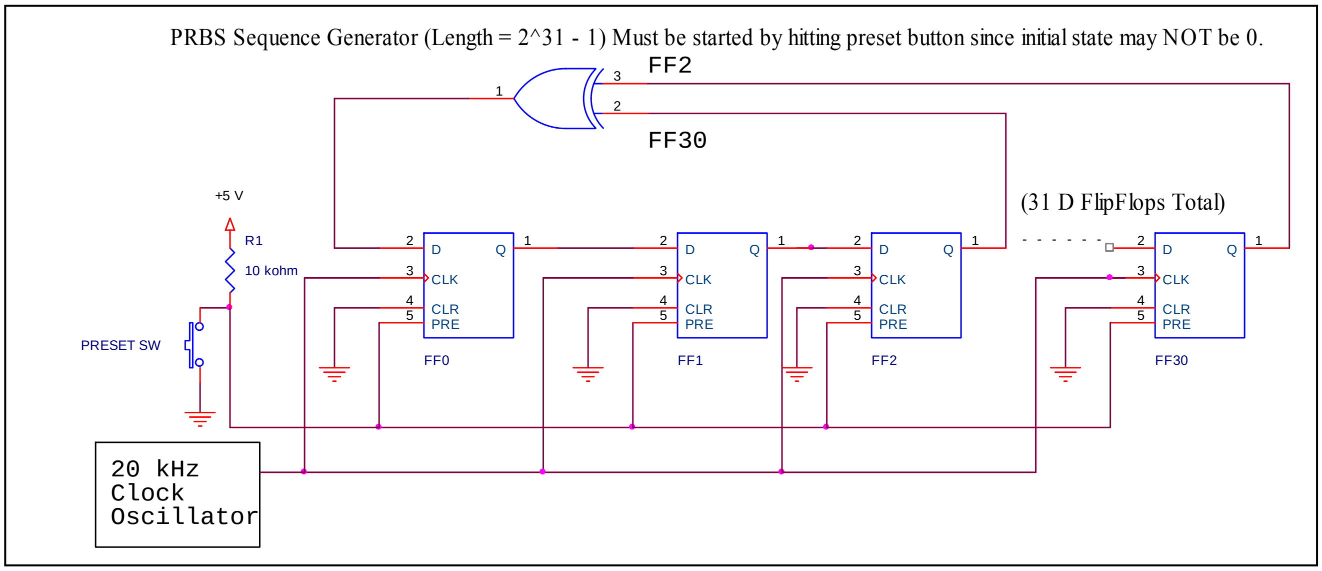
PRBS Sequence Generator (Length = 2^31 - 1) Must be started by hitting preset button since initial state may NOT be 0.
FF2
FF30
(31 D FlipFlops Total)
+5 V
R1
10 kohm
PRESET SW
D
Q
CLK
CLR
PRE
FF0
FF1
FF2
FF30
20 kHz Clock Oscillator

Figure P1. Pseudorandom Binary Sequence Generator – Hardware Implementation

Below is a C-language program written to run on our FreeScale CSMB9S12C128, and it emulates this hardwired white noise generator in software as an interrupt routine. The calling program first calls the C routine TO24MHZ( ) that initializes the PLL to change the bus clock from 2 MHz to 24 MHz. It then calls PRBS_INIT( ) that sets up the TC0 output compare channel to interrupt at a rate of 20 kHz, and it also initializes the flip-flops to all 1's, which is the function performed by the switch PRESETSW in the diagram of Fig. P1. Each TC0 output compare interrupt corresponds to a single 20 kHz clock pulse in the hardwired system above. The main program also enables interrupts before it falls into an idle loop. The interrupt routine PRBS_TC0_ISR emulates the 31 flip-flop circuit that is shown in Fig. P1. *Note that this software emulation should behave exactly like the hardware system in Fig. P1.*

The long unsigned integer variable "shiftreg" in the program below is used to implement the 31-bit shift register in Fig. P1. Flip-flop FF0 corresponds to Bit #0 (the LSB) of "shiftreg", and FF30 corresponds to Bit #30 of "shiftreg". Note that the MSB (Bit #31) of the shift register "shiftreg" is not used.

The system output is taken from FF30, and this output is driven onto output pin PT0. Recall from Lab 5 that if the piezoelectric buzzer (loudspeaker) "BZ" jumper is installed on the Project Board, this buzzer is connected to PT0 on the CSMB9S12C128 board, and when this program is run, we will hear the broadband white noise as a steady "hiss".

Fill in the 20 missing blanks in this C-language program.

```
#include <hidef.h>      /* common defines and macros */
#include <MC9S12C128.h>     /* derivative information */

void PRBS_INIT(void);
void TO24MHZ(void);
interrupt void PRBS_TC0_ISR(void);
long unsigned int shiftreg;
void main(void) {
  TO24MHZ( );
  PRBS_INIT();
  EnableInterrupts;
  for(;;) {} /* wait forever */
}

void PRBS_INIT(void)
  {   DDRT_DDRT0 = ______;                    //Blank 1
      TSCR1_TEN= _________;                   //Blank 2
      TSCR2 = 0b001;
      TC0 = TCNT+ _________;                  //Blank 3.  We desire a 20 kHz interrupt rate
      TFLG1 =         _________;              //Blank 4
      TIOS =          _________;              //Blank 5
      TIE =           _________;              //Blank 6


      shiftreg = 0x7FFFFFFF;                  //Preset all 31 FFs to 1
  }

void TO24MHZ(void)
{
   CLKSEL = CLKSEL & ___________;         //  Blank 7.  Disengage PLL from system
   PLLCTL = PLLCTL |   ___________;       //  Blank 8.   Turn on PLL
   SYNR = _____________;                  //  Blank 9
   REFDV = ____________;                  //  Blank 10.  Set for 24 MHz Bus Clock
  while (!(CRGFLG ________));                    //  Blank 11. Wait till PLL Locks
  CLKSEL = CLKSEL |  ____________;               // Blank 12. Engage PLL into system

}


interrupt void PRBS_TC0_ISR(void)
 {
   char FF0, FF2, FF30;
   TFLG1 = ___________________;                 //Blank 13
   TC0 = TC0 + _________________;               //Blank 14
   shiftreg = shiftreg<<___________;            //Blank 15
   FF2 = (shiftreg>>________) & 1;              //Blank 16
   FF30 = (shiftreg>> _______) & 1;             //Blank 17
   FF0 =  FF2 ______ FF30;                      //Blank 18
   shiftreg = shiftreg + _________________;     //Blank 19
   PTT_PTT0 = ______________________;           //Blank 20
 }
```

2) ***(20 points, 1 point per blank) Assembly Language Program: Interrupt-Driven Optical Theremin***
An optical Theremin is to be built using the CSMB9S12C128 board. This device plays continuously variable musical tones whose frequency is regulated by waving your hand over it. At first this might appear magical to an observer! But in reality, the frequency is varied using a variable voltage that is derived by connecting a light-variable resistor (CdS cell) in series with a 4.7 kΩ fixed resistor. The CdS cell is connected between Vcc = +5 V and the microcontroller's analog input voltage pin AN0 = PTAD0. The fixed 4.7 kΩ resistor is connected between the AN0 pin and ground, thus forming a variable voltage divider. Assume that the resistance of the CdS cell varies between 400 Ω (full light hitting it) and 10 kΩ (very little light hitting it). As the hand waves over the CdS cell, interrupting the amount of light that gets to the CdS cell, the resistance can be varied over this range, and thus you can determine the corresponding AN0 voltage range. (You may safely assume that the input resistance of the AN0 input pin is much, much greater than 10 kΩ.)

Fill in the the blanks in the following code to make the piezobuzzer (connected to pin PT0) generate an audio tone that varies between 110 Hz, which is two octaves below middle A on the piano (CdS cell resistance = 10 kΩ), and 440 Hz, which is middle A on the piano (CdS cell resistance = 400 Ω). PT0 is to be set up to automatically toggle each time an output compare event occurs on timer channel TC0.

```
; export symbols
        XDEF Theremin           ; export 'Entry' symbol
        ABSENTRY Theremin       ; for absolute assembly
        INCLUDE 'MC9S12C128.inc'
        ORG $4000
const32:
        dc.l  __________________   ; BLANK 0
                                   ; dc.l allocates a 32-bit "long word" constant into 4 bytes of flash mem.
                                   ; You must figure out what this constant is in order to properly convert
                                   ; the A/D converted sample into the desired timer delay value necessary
                                   ; to obtain the specified frequency behavior (440 Hz for no light, and
                                            ; 110 Hz for full light) of the Theremin.

Theremin:   lds   #$1000       ; initialize the stack pointer
        movb #________________,TSCR2          ; BLANK 1.  Remember, our bus clock is 2 MHz
        movb #________________,TSCR1          ; BLANK 2.  Start TCNT counting
        bset  ________________,1              ; BLANK 3.  Make TC0 output compare
        bset __________________,1             ; BLANK 4.  Make TC0 interrupt on compare
        movb #________________,TCTL2          ; BLANK 5.  Make PT0 toggle on o.c. event
        bclr DDRAD,1
        bclr ___________________,1             ;BLANK 6.  Make AN0 an analog input
        bset ___________________               ;BLANK 7.  Power up A/D
        ldx #$ffff
wt_for_ATD_power_up:
        _______________________               ;BLANK 8.
        bne wt_for_ATD_power_up               ;Wait for A/D to power up
        movb #%00001000,ATDCTL3
        movb #%00000001,ATDCTL4
        movb #%____________________________  ;BLANK 9.  Start conversion on AN0, Right
                                                       ;justify results in result register.
wt_done:    brclr ____________________________  ;BLANK 10. Wait till conversion done.
        ldd _________________________________  ;BLANK 11.  Put 10-bit result in Accum D
        ldy #5532
        ____________________________________  ;BLANK 12
        subd const32+2
        tfr d,x
        tfr y,d
        bcc _________________________________  ;BLANK 13
        subd #1                               ;Implements 32-bit subtraction
skip_adjust:
        subd const32
        tfr d,y
        tfr x,d
        ldx #1000
        ____________________________________  ;BLANK 14
        tfr y,d
        addd TCNT
        std  _________________________________  ;BLANK 15 Schedule first o.c. interrupt
        movb _______________________________  ;BLANK 16 Clear interrupt flag
        cli
idle_loop:  bra idle_loop


tc0isr:     movb ______________________________  ;Not graded
wt_done1:   brclr _____________________________  ;Not graded
        ldd ATDDR0
        ldy #5532
        ____________________________________  ;Not graded
        subd const32+2
        tfr d,x
        tfr y,d
        bcc ________________________________        ;Not graded
        subd #1
skip_adjust1:
        subd const32
        tfr d,y
        tfr x,d
        ldx #1000
        ___________________________________         ;Not graded
        tfr y,d
        addd TC0
        ___________________________________         ;BLANK 17
        movb #1,___________________________         ;BLANK 18
        rti

        ORG   $FFFE
        DC.W  Theremin          ; Reset Vector
        ORG   _____________________________         ;BLANK 19
        DC.W  tc0isr
```

3) **(12 points) LCD Display Multiplexing**

a) A custom LCD display for a new product has 300 segments that must be individually controlled (turned on or off). If we choose to use 1:4 multiplexing on this display, implying 4 back plane signals are needed, what is the total number of wires (back plane wires *plus* front plane wires) that must be connected to this display?

Total # Wires = ____________

b) Repeat Part A for 1:7 multiplexing.

Total # Wires = ____________

c) For the case of 1:7 LCD multiplexing, there are 7 backplane signals, BP1, BP2, BP3, BP4, BP5, BP6, and BP7. Assume that Vcc = 5 V, so the waveform voltage levels are 5 V, 3.333 V, 1.666 V, and 0 V. Sketch one frame of the **BP2** backplane signal and also one frame of the **BP3** backplane signal.

d) Sketch one frame of a single front plane signal, FP1, where the segments that pass over BP3, BP5, and BP6 are to be ON, and the remaining four segments are to be OFF.

e) Sketch one frame of the voltage waveform $Vseg_{31}$, which represents the voltage across the "turned ON" segment that lies between FP1 and BP3. ($Vseg_{31}$ = BP3 voltage – FP1 voltage). Use the FP1 voltage waveform from Part d above.

f) Sketch one frame of the voltage across the "turned OFF" segment that lies between FP1 and BP2, $Vseg_{21}$. ($Vseg_{21}$ = BP2 voltage – FP1 voltage) . Use the FP1 voltage waveform from Part d above

g) Find the RMS value of the $Vseg_{31}$ waveform of Part e, which corresponds to the waveform of a turned **ON** segment, and also the RMS value of the $Vseg_{21}$ voltage waveform of Part f, which corresponds to a turned **OFF** segment. For credit on this problem, you ***must*** show the steps in your calculation (not just write down numbers) in the space below.
Recall that in the class notes, it was shown (in Figure 7.21) that for the case of 1:4 multiplexing, the RMS voltage across a segment that is ON is Vrmson = 2.899 V,rms; and the RMS voltage across a

segment that is OFF is Vrmsoff = 1.67 V, rms.

RMS value of $Vseg_{31}$ = __________ V,rms   RMS value of $Vseg_{21}$ = _________ V,rms

h) Based upon comparing the results for 1:4 and 1:7 multiplexing,

(a) which multiplexing method requires fewer connections? ___________________

(b) which multiplexing method yields higher contrast? ___________________

4) (5 pts) An NPN power BJT with a $\beta = 100$, a forward BE junction voltage drop of 0.7V, and a Vce(sat) = 0V is used to switch ON and OFF a 3 Ω, 12 V (48 Watt) resistive load using a circuit similar to the ***upper-left circuit of Slide #57***. (Assume the power supply connected to the load is now 12 V.)

a) ***Draw this circuit in the space below***, and then determine the maximum permissible value of Rb that will still keep the BJT ***saturated*** while the load is ON. *Note that with the BJT saturated, the switching BJT consumes essentially NO power ($P_{BJT} = Ic*Vce = 4*0 = 0W$), and the load receives the full $P_{LOAD} = I_L*V_L = 4*12 = 48$ W from the dc power supply.*

$Rb_{(MAX)}$ = _____________

b) How much current must the open-collector driving gate be able to sink while the load is turned off? (Assume that the value of Rb is the value calculated above in Part a, and that the output voltage of the open-collector driving gate is 0.3 V when sinking this current.)

$Iout_{SINK}$ = _________

c) If Rb = 500 Ω, and the open-collector driving gate is switched to its HIGH (floating) state, find the power that is delivered to the (3 Ω, 12 V, "48 Watt") load
(*Hint: Because Rb = 500 Ω violates the calculation in Problem 4(a), you will find that the power delivered to the load will far less than the desired 48 Watts!*
Also find the power that is dissipated (as heat) in the BJT switching transistor. *(Hint: you may ignore the small amount of power consumed in the base-emitter junction of the BJT, and so assume that P_{BJT} = Vce * Ic.*

Because Rb = 500 Ω violates the calculation in Problem 4(a), the power consumed (as heat) in the switching transistor will be unacceptably large, and it may even burn out the switching transistor! Also, the power delivered to the 48 W load is unacceptably small!)

$P_{3\Omega\ LOAD}$ = _______________

P_{BJT} = _______________

5) (4 points) Imagine that the 3 Ω resistive load of Problem 3 is replaced by an inductive load that may be modeled as a ***<u>1.0 H inductance in series with a 3 Ω resistance</u>***.

a. Sketch the modified switching circuit in the space below. This circuit consists of the open-collector driving gate, the 12 V dc power supply, Rb (assume Rb has a value that is less than the maximum value calculated in Problem 4(a)), the switching BJT, and the load (1 H inductor in series with a 3 Ω resistor).

b. Assume that the open-collector driving gate output voltage has been LOW for a long time, and then it suddenly is raised to its HIGH (floating) state. How long after that will it take for the load current to reach 90% of its final value (3.6 Amperes)? Let us regard this as the load "turn-on" time. (***<u>Hint: Study Lecture 11, Slides 71-78</u>***)

Load Turn-On Time = _____________

6) (6 pts) Now imagine that the driving gate output voltage of the circuit in Problem 5 is suddenly changed from HIGH to LOW. ***<u>Hint: See Lecture 11, Slides 71-78</u>***

a) Using $v_L = Ldi_L/dt$ to explain why the switching BJT could burn out.

b) Redraw the switching circuit showing how a ***<u>single</u>*** fast-acting diode may be added to this circuit to solve the problem of Part (a).

c) For the circuit of Part (b), determine how long it will take for the load current to decay from its full value down to 10% of this value (0.4 A) when the driving gate output voltage is suddenly changed from HIGH to LOW . You might regard this as the "load turn-off time". Assume the ON resistance of the diode is negligible. ***<u>Hint: See Lecture 11, Slides 71-78</u>***

Load Turn-Off Time = _____________

d) How could you make this load turn off time shorter? Redraw the circuit showing how one additional resistor "Rspeedup" might be added, so the load will turn off faster, without affecting the load current of the load turn-on time. If Rspeedup = 10 ohms in this example, calculate the new load turn-off time.

New Load Turn-Off Time = _____________

7) (6 pts) Using only ***TWO*** rising-edge sensitive D flip-flops (with D, CLK, CLR, Q and Q\ pins) and assorted inverters and other logic gates, design a circuit that will produce the ***2x resolution*** ***CW output waveforms*** shown in Fig. 6 from the A and B input waveforms. (See arrow below that points to the two "CW" waveforms that you are to design your circuit to produce.) YOU NEED NOT DESIGN THE CCW output detection circuit. ***Be sure to label your circuit's A and B inputs as well as your circuit's CW output.***

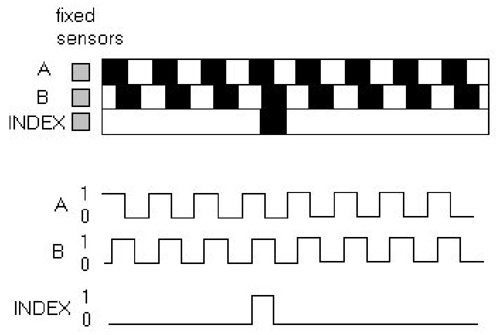

Fig 5. Incremental encoder disk track patterns

Desired output waveform→

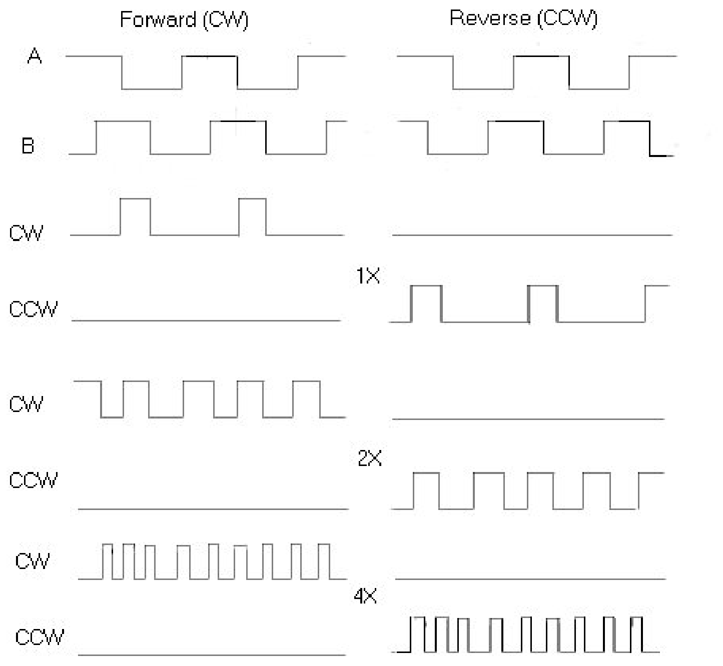

Fig 6. Quadrature direction sensing and resolution enhancement. (CW = clockwise, CCW= counter-clockwise)

8) ***Stepping Motor (6 points)***
Referring to the stepping motor circuit diagram shown in the course notes (Slide #68), imagine that the two bottom rows of 7407/7406 inverters are removed, leaving us with just one row of 2N6427 power Darlington BJT transistors. Then imagine that a microcontroller has PA3 (Port A, Pin 3) connected to the base of the left-most power Darlington, PA2 to the next one, PA1 to the next, and finally PA0 to the right-most power Darlington.

a) List the sequence of eight 4-bit numbers that would have to be output on the low 4 bits of PORT A (in the order PA3:PA2:PA1:PA0) in order to make the magnetic field vector developed by the stepping motor stator coils step in the clockwise (CW) direction, with 8 steps per revolution (45 degrees per step). Let your first number correspond to the magnetic field pointing directly up. (Hint: you may turn on ***<u>either 1 or 2 coils</u>*** at a time.)

________, ________, ________, ________, __________, _______, _______, ________

b) Assuming a permanent magnet rotor with 9 permanent magnet poles (instead of the rotor with 3 permanent magnet poles considered on Slide #70 in the lecture notes), determine the number of steps per revolution of the shaft using the 8-value sequence of Part A.

Do this by drawing, in the space provided below, the 9-pole rotor (showing only the 9 equal-angularly spaced south poles) with one of the 9 poles aligned with the initial **B** field. Then, when the **B** field steps 45 degrees to its next position, determine which south pole is closest to the new position of the **B** field, and hence is pulled into alignment.

Determine the angle through which the shaft rotates, and determine its direction of rotation (CW or CCW). Also determine the total number of steps per one 360 degree revolution of the shaft.

*<u>Drawing of 9-pole Permanent Magnet with one pole aligned with initial **B** field.</u>*

Degrees of Shaft Angle Rotation Per Step = ____________

Step Direction = ______________

Number of steps per one 360 degree revolution of the shaft = __________

9) (1 pt) What is the best name for the four 1N4001 power diodes in this stepping motor circuit? (circle one)
1. transient voltage suppression diodes 2. turn-on speedup diodes 3. turn-off speedup diodes
4. load current limiter diodes

a. (1 pt) What is the best name for the 22-ohm resistor in this stepping motor circuit? (circle one)
1. turn-on speedup resistor 2. turn-off speedup resistor 3. load current limiter
4. voltage transient suppression resistor

10) (2 pt) A magnetic reed switch will be most sensitive to an applied magnetic field (**B**) that is oriented in a direction that is
1. perpendicular to the reeds 2. parallel to the reeds 3. at a 45 degree angle to the reeds

11) (2 pt) What is the purpose of the diodes in the 8 x 8 scanned keyswitch matrix discussed in the course notes?
1. short-circuit protection 2. over-voltage protection 3. speed up key scanning process

12) (9 pts) Imagine that a "poor man's A/D" circuit implemented in the C language is used to sense the value of a variable resistor Rx by connecting Rx between PT0 and Vcc = 5.0 V and a 0.33 μF capacitor between PT0 and ground. Assume that PT0 (when configured as an input) has an input logic high threshold of 3.00 V. If PT0 (when configured as an output) is driven low (to 0 V) for several seconds, and then suddenly released (allowed to float), the time elapsed before a logic 1 level is read by the microcontroller is measured.

a) Find the value of Rx if the time elapsed before a logic 1 is read is found to be 5 ms?

b) Find the value of Rx if the time elapsed before a logic 1 is read is found to be 10 ms?

c) What is the lowest value of Rx that can be measured using this scheme if PT0 cannot sink more than ***25 mA*** when driving its output to a logic 0 level (which we will assume is precisely 0 V).

d) How should the LSB of the PERT register be set in order to obtain the most accurate measurement of Rx? Explain your reasoning.

e) How would you set the LSB's of the Port T Data Register and the Port T Data Direction Register in order to drive PT0 to 0 V?

f) How would you set the LSB's of the Port T Data Register and the PORT T Data Direction Register in order to release (float) PT0?

13) (7 pts) **UPC-A Bar Code** (Used on groceries, pharmaceuticals, electronic items, but NOT on books!)

a. Using the UPC-A encoding table found in the notes, determine the six encoded UPC digits in the ***left half*** of the bar code. Recall that Black = 1, White = 0; there are 3 SYNC patterns: 101 at each end, and 01010 in the middle. (*Hint: first make sure you can successfully decode the six left digits in the example UPC code in the notes, or on any grocery product in your home.*)

b. Recalling that the UPC-A encoding table found in the notes must have its ***black and white regions exchanged*** for the ***right half*** of the UPC code, determine the six encoded UPC digits in the right half of the bar code. (*Hint: first make sure you can successfully decode the six right digits in the example UPC code given in the notes.*)

c. The last (rightmost) digit you found in Part (b) is the UPC-A checksum digit. In the space below, show the step-by-step calculation of this checksum digit from the other preceding 11 digits. Your results must match the 12th digit you decoded above.