UNC Charlotte–ECGR4101/5101-Midterm Exam –10/05/06 Solution

1. E
2. A
3. A - see notes page 4-6 (4-19)
4. E - see notes page 4-6 (4-19)
5. C
6. D
7. D
8. A
9. B
10. B
11. D
12. D
13. E - the best answer is that there is not enough voltage
14. C - SP points to the top of the stack, not to temp variable space
15. B
16. C - most of you forgot about the 4k of EEPROM-mapped space
17. D
18. C
19. D
20. B
21. D
22. A
23. C
24. D - I reviewed wire bond in great detail in class.
25. A

Problem 26

Range: xA0000 to xFF8FF 2 points
= x5F900 = (65536*6) – (4096) + (9*256) = 391424 bytes 3 points – must not be 382k!!!

Problem 27

$$n = \left[\frac{(V_{in} - V_{-ref})(2^N - 1)}{V_{+ref} - V_{-ref}} + 1/2 \right]_{\text{int}}$$

But step size is simple (5v – (-5v))/ 2^N = 10v/256 = 0.039v (Acceptable answers are 10v/256, 0.039v, 0.04v) 5 points, 3 for method, 2 for answer

00110010 = 32+16+2 = 50, (50*10v)/256 – 5v = 500v/256 – 5v = -3.047v
About -3.0 v 5 points, 3 for method, 2 for answer

Problem 28

```
// Name: James Conrad - 10/05/06
// Function: when sw1 is pressed, turn the red LEDs on
// If the thermister value is above 127 light the yellow LED
// If the light sensor value is above 127 light the green LED
// Inputs: sw1, 2 ADC;   Outputs: LEDs
Setup a switch (input)
Setup LEDs (output, turn off)
Setup thermister and light sensor ADC
Start continually reading thermister, light sensor
While (1) {
   If (sw1 pressed) Turn on the red LED
         Else turn off red LED;
   If (thermister>127) Turn on the yellow LED;
         Else turn off yellow LED;
   If (light sensor>127) Turn on the green LED
         Else turn off green LED;
   }
```

Points:

5 points: header comments
5 points: set up switch, LED, ADC
5 points: While loop with testing values inside (continuously)
5 points: Turn on LED with conditions
5 points: Turn off LED with condition
No extra points for writing the entire code package. I only asked for the algorithm.

Problem 29

```
adcon0=0x18;       // perform repeated A/D conversions – sweep
adcon1=0x29;       // sweep pins 0-3 (to get pins 1 and 2)
adcon2=0x01;       // sample and hold
adst=1;            // <- start measuring!
```

Points:

5 points: set up control regs (adcon0, 1, 2)
3 points: set up regs with correct values
2 points: start sampling (adst or adcon bit set)
No extra points for writing the entire code package. I only asked for a few lines.

FYI:

```
adcon0 = 0x18;
      /* xx011xxx;     /* AN0 input, 1 shot mode, soft trigger
         ||||||||______analog input select bit 0 – don’t care
         |||||||_______analog input select bit 1 – don’t care
         ||||||________analog input select bit 2 – don’t care
         |||||_________A/D operation mode select bit 0 - sweep
         ||||__________A/D operation mode select bit 1 - sweep
         |||___________trigger select bit - software
         ||____________A/D conversion start flag – set any time
         |_____________frequency select bit 0 – don’t care */

      adcon1 = 0x29;
/*      xx1x1001;       ** 10-bit mode, Vref connected
        ||||||||______A/D sweep pin select bit 0 – 1 for pins 0-3
        |||||||_______A/D sweep pin select bit 1 – 0 for pins 0-3
        ||||||________A/D operation mode select bit 1
        |||||_________8/10 bit mode select bit = 1 = 10-bit
        ||||__________Frequency select bit 1 - don’t care
        |||___________Vref connect bit
        ||____________External op-amp connection mode bit 0
        |_____________External op-amp connection mode bit 1 */

      adcon2 = 0x01;
   /*    000x0001;      ** Sample and hold enabled
         ||||||||______AD conversion method select bit
         |||||||_______AD input group select bit 0 – port 10
         ||||||________AD input group select bit 1 – port 10
         |||||_________Reserved
         ||||__________Frequency select bit 2 – don’t care
         |||___________Reserved
         ||____________Reserved
         |_____________Reserved */
```