Real-time programming

Lecture questions Part2

L5 Interrupts

1. Explain the concept of interrupts. Give some examples.
2. Explain the difference between subroutine call and interrupt service routine.
3. Compare polling method against interrupt method for communication with an peripheral.
4. Compare maskable and non-maskable interrupts.
5. Explain interrupt service routine sequence. What steps are required when an interrupt request is to be serviced?
6. What methods are used to identify interrupt requests? Briefly explain each of them.
7. In case of multiple interrupt requests, how interrupt priorities could be handled?
8. What is an external interrupt and what is pin change interrupt?
9. Name at least 5 interrupt sources for ATMega328.

L6 Debouncing and Timers/Counters

1. Explain bouncing problem and debouncing techniques in detail.
2. Explain timer operation principle.
3. What is the difference between timer and counter mode?
4. Draw and explain basic timer structure.
5. Explain timer operation without comparator.
6. Explain timer operation with comparator.
7. Explain all methods for counting range extension.
8. Fosc=X Hz, division factor= p. If the timer is reset after it counted for Y pulses, what is the time interval between two resets? (X, p and Y will be actual numbers on exam)
9. Explain watchdog timer operation.
10. Give and explain one example where timers are used for PWM.
11. Explain interrupt sources in interrupt vector table for ATMEga328p timer 0.
12. Explain ATMega328p timer 0 comparator module operation.

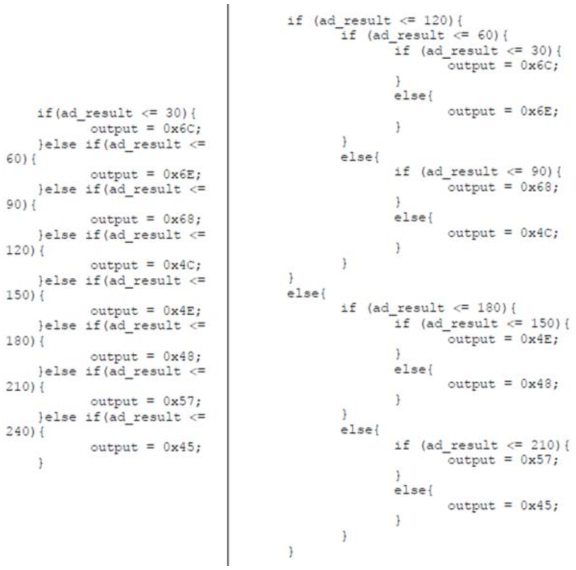
L7 RTOS

1. What are real-time systems and what are the main characteristics of real-time systems?
2. What is the Real-Time Operating System (RTOS)? Explain the structure of RTOS.
3. Explain RTOS classification and give a few examples. What are the differences?
4. What is a task? Explain multitasking in RTOS.
5. Name and briefly explain RTOS types.
6. Task states. Show and explain the diagram with task states and task state transitions.
7. Explain stack related problem when using subroutines within RTOS.
8. What is a context-switch? Why is it necessary and how it influences overall system behavior?
9. What type of messages as a part of inter-task communication are used?

10. Explain how semaphores work. What types of semaphores can be used within RTOS?
11. How mutexes work and when are they usually used?
12. Show diagram and explain how task switching based on priority is done (3 tasks with different priorities).
13. Show diagram and explain task switching with time sharing with priorities (3 tasks with the same priority and one with higher priority).
14. What is the problem with round-robin scheduling and how is it solved by using linked lists?

L8 Code optimizations

1. Within code optimizations, what can be optimized?
2. What optimization switches can be used with AVR-GCC compiler?
3. Why is variable size important?
4. Compare loop jamming and loop unrolling in terms of code optimizations.
5. When is inline assembly code used and why?
6. Give an example of bit macro function and usage of bit masks.
7. Why function calls within interrupt service routines are bad practice?
8. Comment the following code fragments efficiency:



```
    if(ad_result <= 30){
            output = 0x6C;
    }else if(ad_result <=
60){
            output = 0x6E;
    }else if(ad_result <=
90){
            output = 0x68;
    }else if(ad_result <=
120){
            output = 0x4C;
    }else if(ad_result <=
150){
            output = 0x4E;
    }else if(ad_result <=
180){
            output = 0x48;
    }else if(ad_result <=
210){
            output = 0x57;
    }else if(ad_result <=
240){
            output = 0x45;
    }
```

```
if (ad_result <= 120){
        if (ad_result <= 60){
                if (ad_result <= 30){
                        output = 0x6C;
                }
                else{
                        output = 0x6E;
                }
        }
        else{
                if (ad_result <= 90){
                        output = 0x68;
                }
                else{
                        output = 0x4C;
                }
        }
}
else{
        if (ad_result <= 180){
                if (ad_result <= 150){
                        output = 0x4E;
                }
                else{
                        output = 0x48;
                }
        }
        else{
                if (ad_result <= 210){
                        output = 0x57;
                }
                else{
                        output = 0x45;
                }
        }
}
```

9. Where constants should be stored and why?