Spisak pitanja iz teorije RSES

1) Objasniti, nacrtati vrste U/I portova mikrokontrolera i uporediti ih.
2) Nacrtati koja je ispravna varijanta povezivanja tastera/prekidača na U/I port open drejn tipa i objasniti.
3) Objasniti zašto je open drejn izlaz bolji u spoju *current sink* nego *current source*.
4) Objasniti ožičenu I logiku.
5) Kod ATmega328 pomoću koja tri registra se upravlja portovima i koji čemu služi?
6) Nacrtati generalnu strukturu portova ATmega328 i objasniti kako se konfiguriše pin porta da bude ulazni/izlazni/sa *pull-up* otpornikom.
7) Zašto je bitno efikasno programiranje na mikrokontrolerima?
8) Koje su prednosti C jezika u odnosu na asembler?
9) Navesti bar jedan primer kada je asembler u prednosti u odnosu na C.
10) Navesti bar 5 razlika programiranja za mikrokontrolere u odnosu na personalni računar.
11) Koji su tipovi izvornih fajlova i koja je razlika među njima?
12) Objasniti razliku između definicije i deklaracije.
13) Čemu služi `extern`? Navesti primer.
14) Čemu služi `static`?
15) Objasniti kompletan postupak projektovanja od izvornih fajlova do izlaznog fajla.
16) Koja 2 tipa podataka se najčešće koriste kod 8-bitnih mikrokontrolera?
17) Navesti neke od ključnih razlika između programera embedded sistema i klasičnih programera.
18) Objasniti razliku između stek metode i metode preklapanja (*overlay*) pri čuvanju lokalnih promenljivih.
19) Nacrtati raspodelu memorije lokalnih promenljivih metodom preklapanja za zadati slučaj.
20) Čemu služi `PROGMEM` direktiva?
21) Gde je preporučeno da se čuvaju konstantni nizovi, stringovi i sl. i zašto?
22) Čemu služi `const` direktiva?
23) Šta je to preprocesor?
24) Šta je i čemu sve služi `#define`?
25) Šta je uslovno prevođenje i koje se direktive koriste za to?
26) Kako su podaci raspoređeni u memoriji?
27) U čemu je osnovna razlika između struktura i unija u programskom jeziku C?
28) Ukoliko su definisane struktura i unija koje sadrže po 3 promenljive (p1, p2 i p3) i koje su sledećih veličina (1, 2 i 4) bajta respektivno, koliko bajtova memorije zauzima struktura, a koliko unija.
29) Objasni pojam *structure padding*.
30) Koja je alternativa povezivanju spoljašnjih periferija umesto standardnog memorijskog mapiranja i kako se radi?
31) Navesti prednosti i mane emulacije memorijskog mapiranja.
32) Navesti, nacrtati i objasniti četiri kategorije ulaznih i izlaznih periferijskih kola.
33) Ako je memorijski mapirano memorijsko kolo izlazni registar bez mogućnosti čitanja stanja, šta je potrebno uraditi za pravilan rad sa takvim kolom?
34) Objasniti pojam RMW komandi u C-u i zokružiti koje su od narednih naredbi RMW tipa.
    x *= y; x = y + 1; x=x+1; x++; x>>=y; x = 1 >> y;
35) Koji tipovi petlji postoje u programskom jeziku C i koliko se minimalno puta izvršava svaka od tih petlji (odnosno komande unutar tela petlje)?
36) Kako se od `FOR` petlje dobija `WHILE` petlja i obrnuto?
37) Navesti sve razlike između `FOR` i `WHILE` petlji.
38) Šta je bolje koristiti u okviru testa kod petlji: predekrement ili postdekrement? Zašto?
39) U kojim slučajevima ima smisla koristiti petlje za generisanje kašnjenja?
40) Zbog čega se izbegava korišćenje petlji za generisanje dugačkih kašnjenja?

41) Ako je ipak potrebno generisati kašnjenje pomoću petlje, kako je to najbolje uraditi?
42) Navesti načine za promenu jednog ili više bita (na nulu, na jedinicu i invertovanje), objasniti ih i navesti po jedan primer.
43) Objasniti primenu *bitwise* maski pri kreiranju binarnog brojača modula 2^n.
44) Objasniti primenu *bitwise* maski u logičkim testovima i zašto je to bolje od upotrebe operatora poređenja.
39) Objasniti skeniranje matrične tastature.
40) Objasniti problem *bouncing*-a.
41) Navesti i objasniti metode *debouncing*-a tastera.
42) Objasniti problem *ghosting*-a i *masking*-a kod rada sa matričnom tastaturom i kako se rešava.
43) Ako je matrična tastatura povezana na port P0, objasniti šta i kako radi sledeća funkcija:

```
unsigned char GetKeyPressed(){
   unsigned char r,c;
   P0 = 0xFF;
   for(r=0;r<4;r++) {
      P0 &= ~(0X10<<r);
      for(c=0;c<4;c++){
         if(!(P0 & (0X01<<c)))
            return (r*4+c);
      }
   }
   return 0xFF; //nije pritisnut ni jedan taster
}
```

44) Koje su osnovne razlike između obične i prekidne funkcije?
45) Koja je razlika između direktne i odložene obrade prekida. Kada se svaki od načina primenjuje?
46) Za datu prekidnu rutinu tajmera koja realizuje periodični interval označiti da li je u pitanju direktna/odložena obrada i da li je interval unutar/van hardverskog opsega tajmera.
47) Koji se problem može javiti u slučaju promene vrednosti globalne promenljive unutar prekidne rutine i kako se taj problem rešava? Navesti primer.
48) Objasniti koji se problem može javiti ukoliko su promenljive veće od jednog bajta (prilikom pojave prekida) i kako ga rešiti?
49) Objasniti pojam ‘Super-petlja’.
50) Šta se podrazumeva pod metodom *Round Robin*?
51) Navesti prednosti i mane izvršavanja programa u super-petlji.
52) Objasniti pojam događaja (eng. *Event*) i način njegovog korišćenja.
53) Šta je softverski FIFO bafer, kada i kako se koristi?
54) Navesti i ukratko objasniti sva tri načina realizacije softverskih FIFO bafera.
55) Šta je softverska mašina stanja?
56) Objasniti suštinu pojma pseudo-paralelni rad kod softverskih mašina stanja.
57) Na koji način se više funkcija realizovanih u formi beskonačne petlje mogu zameniti mašinama stanja?
58) Na koji način se standardne C petlje mogu implementirati u okviru mašine stanja?
59) Objasniti sledeće pojmove: *Event, Semaphore, Message, Mailbox.*
60) Navesti i objasniti osobine pisanja programa u formi mašine stanja u odnosu na operativne sisteme.
61) Šta je task?
62) Koja su moguća stanja taska? Objasniti.
63) Šta predstavlja multitasking operativni sistem?
64) Dati grafički prikaz jednostavnog RTOS sa dva taska i objasniti funkcije OS.
65) Kako se klasifikuju RTOS u zavisnosti od vremenskih ograničenja na odziv? Dati primere.
66) Kako se klasifikuju RTOS sa stanovišta promene aktivnog taska?
67) Koje su osnovne razlike između Cooperative i Preemptive operativnih sistema?

68) Navesti razlike pri primeni mašine stanja i multitasking operativnog sistema.
69) Navesti potencijalne probleme steka u kooperativnom sistemu i moguća rešenja.
70) Navesti osnovne prednosti i mane u primeni multitasking operativnih sistema u odnosnu na klasične načine pisanja programa.
71) Objasniti pojam *Dead-Lock*?
72) Šta je Preemptive multitasking operativni sistem?
73) Zbog čega taskovi u Preemptive operativnom sistemu moraju imati sopstvene stek memorijske blokove, odnosno zbog čega se ne koristi jedinstven stek?
74) U kom slučaju je veća potrošnja RAM memorije, u Cooperative ili Preemptive načinu rada i zašto?
75) Ako su, u Preemptive operativnom sistemu sa Round Robin mehanizmom, taskovi različitih prioriteta, da li se može dogoditi da se neki od taskova nikad ne poziva? Objasniti.
76) U slučaju da problem iz prethodnog pitanja postoji, na koji način se on može prevazići?
77) Da li je, u Preemptive operativnom sistemu, moguće napisati program sa više taskova koji pravilno rade, iako ni u jednom trenutku ne pozivaju neku od sistemskih funkcija za promenu aktivnog taska? Objasniti.
78) Šta su, kako funkcionišu i čemu služe *mutex*-i?
79) Nabrojati i ukratko objasniti mehanizme komunikacije među taskovima.
80) Navesti bar 4 karakteristike “kulturno” pisanih C programa i objasniti ih.
81) Navesti neke od razloga i objasniti zašto je tema “kulturnog” programiranja važna.
82) Objasniti pravilo “4 space-a”.
83) Objasniti značaj vertikalnog i horizontalnog formatiranja koda.
84) Diskutovati korisne/nekorisne komentare.
85) Za sledeća 3 literala u C programskom jeziku napisati šta skoro sigurno predstavljaju (varijabla/konstanta/ime funkcije):
MAX_COUNT:_____; MaxCount:_____; max_count:_____
86) Komentarisati dati primer koda sa stanovišta "(ne)kulture" programiranja.
87) Objasniti šta podrazumevamo pod pojmom optimizacija koda?
88) Objasniti AVR-GCC optimizacione prekidače.
89) U vezi optimizacije koda objasniti sledeći ključni savet: _____.