Studijski program D (Računarstvo i informacione tehnologije). **Principi programiranja**. Ogledni primjeri pitanja za Završni ispit (50 poena). Od naslova 1. Primjeri logičkih elemenata u digitalnoj elektronici do naslova 38. Naknadno dodato: jednostavni primjeri za ulaz–izlaz. Dolaze 3 pitanja iz teorije + 2 zadatka (programi na jeziku asemblera za Intelove procesore).

1. Grafički simboli za logičke elemente AND, OR, NOT, NAND, NOR i XOR. Odgovarajuće jednačine.

2. Sabirač (engl. semi–adder) $z = z(x, y)$, $q = q(x, y)$: definicija, tablica, jednačine i mreža. Isto se kaže i polu–sabirač.

3. Potpuni sabirač (engl. full adder) $z = z(x, y, p)$, $q = q(x, y, p)$: definicija, tablica, jednačine, mreža i grafički simbol.

4. Definicija paralelnog sabirača (engl. parallel adder) $x_4x_3x_2x_1 + y_4y_3y_2y_1 = z_4z_3z_2z_1$. Konstrukcija paralelnog sabirača pomoću četiri potpuna sabirača.

5. Dekoder sa dva ulaza i četiri izlaza $y_1 = y_1(x_1, x_2)$, $y_2 = y_2(x_1, x_2)$, $y_3 = y_3(x_1, x_2)$, $y_4 = y_4(x_1, x_2)$: definicija, jednačine, tablica i šema.

6. Dekoder sa tri ulaza i osam izlaza $y_1 = y_1(x_1, x_2, x_3)$, $y_2 = y_2(x_1, x_2, x_3)$, ..., $y_8 = y_8(x_1, x_2, x_3)$: definicija, jednačine i tablica.

7. SR leč kolo: pojam (ima dva ulaza S i R i ima izlaze Q i $\overline{Q}$), šema, tabela, jednačina $Q_{n+1} = Q_{n+1}(Q_n, S, R)$ i grafički simbol. Nedozvoljena kombinacija na ulazu.

8. SR leč kolo sa signalom dozvole (ima tri ulaza S, R i C i ima izlaze Q i $\overline{Q}$): pojam razmatrane mreže (definicija razmatrane mreže), tabela, jednačina $Q_{n+1} = Q_{n+1}(S, R, C, Q_n)$ i grafički simbol. Signal dozvole označava se kao C.

9. Stacionarni registar: njegov pojam tj. njegova definicija, radnja ili svejedno funkcija koju obavlja i grafički simbol.

10. Pomerački registri: definicija, grafički simbol, primjena za aritmetičke operacije i primjena za serijski prenos informacija.

11. Nabrojati i opisati glavne registre osnovnog računara (PC, ..., AC).

12. O podjeli naredbi osnovnog računara na tri vrste.

13. Šta je to obično (neposredno) adresiranje a šta je posredno (indirektno).

14. Vremenski ciklusi osnovnog računara (mašinski ciklusi).

15. Šema kontrolne jedinice osnovnog računara i objasniti.

16. Mikro–operacije prijemnog ciklusa (osnovni računar).

17. Registri za ulaz–izlaz i prekide (osnovni računar).

18. Primjeri ulaznog odnosno izlaznog prenošenja koje je programski kontrolisano (osnovni računar).

19. Rutina za obradu prekida R i dva bafera u memoriji (osnovni računar).

20. Primjer izlaznog prenošenja koje je prekidom vođeno (osnovni računar).

21. Sastaviti program za osnovni računar za računanje vrijednosti izraza $y = 2a+b+1$. Aritmetički program.

22. Rad sa potprogramom. Sastaviti program za osnovni računar za računanje vrijednosti izraza $y = (a/16 + b)/16$, gdje je prisutan potprogram za dijeljenje sa 16.

23. Apsolutni loader L (osnovni računar).

24. Loader koji vrši relokaciju (osnovni računar).

25. Realizacija sabiranja i množenja u slučaju nepokretnog odnosno pokretnog zareza (uopšte).

26. Organizacija ulazno–izlazne jedinice (uopšte).

27. Glavni registri procesora Intel 8086 (AH, ..., IP), obrazovanje fizičke adrese po formuli $a = 16s + d$.

28. Registar PSW procesora Intel 8086 tj. flagovi.

29. Aritmetički program: sastaviti program na jeziku asemblera za računanje četiri broja $y = 2ab + 1$, $y = y - 2$, $y = y - 2$ i $y = y - 16$ (primjera).

30. Rad sa nizom: sastaviti program na jeziku asemblera za sabiranje svih članova jednog niza brojeva (primjerd).

31. Smisao naredbi MOV, PUSH, POP, IN, OUT, PUSHF i POPF (služe za prenos podataka, Intel 8086).

32. Smisao naredbi ADD, ADC, SUB, SBB, CMP, INC i DEC (aritmetičke naredbe, Intel 8086).

33. Metode adresiranja: immediate (kada piše n), direct (kada piše $[n]$), register (kada piše AX ili ...) i indirect (kada piše [BX] ili [BP] ili [SI] ili [DI]).

34. Asemblerske direktive DB immed i DW immed (kod debug.exe). Define byte, define word.

35. Sistem prekida procesora Intel 8086: samo o prekidima od $n = 0$ do $n = 4$ i o tri moguća izvora–uzroka da dođe do prekida.

36. O mikroprocesoru Intel 80286.

37. O mikroprocesoru Intel 80386.

38. O mikroprocesoru Pentium: četiri generacije Pentiuma, tekuća traka (pipeline) i dohvatanje unaprijed (prefetch).

39. Napisati program na jeziku asemblera (da se izvrši pomoću debug.exe) za računanje izraza $y_1 = 4a + 4b + 1$ i $y_2 = 4y_1 + 4$, gdje su a i b dati brojevi, recimo $a = 24$ i $b = 28$, ovo je u dekadnom. Svi brojevi su veličine po dva bajta. Pridružiti promjenljivima a, b, y_1 i y_2 adrese (programer treba da napravi raspored).

40. Sastaviti program na jeziku asemblera (da se izvrši pomoću debug.exe) za računanje četiri broja: $y_1 = 2a + 2b - 2$, $y_2 = y_1 + 4$, $y_3 = y_2 + 4$ i $y_4 = y_3 + 32$, gdje su a i b dati brojevi. Svi brojevi su veličine po dva bajta. Programski zadati $a = 16$ i $b = 32$ (ovo je u dekadnom). Drugim riječima, vrijednosti a i b treba da budu upisane u memoriju pomoću naredbe MOV.

41. Sastaviti program na jeziku asemblera (da se izvrši pomoću debug.exe) za računanje četiri broja: $y_1 = 2a + 2b + 2$, $y_2 = y_1 - 4$, $y_3 = y_2 - 4$ i $y_4 = y_3 - 32$, gdje su a i b dati brojevi. Svi brojevi su veličine po dva bajta. Programski dodijeliti vrijednosti $a = 16$ i $b = 32$ (ovo je u dekadnom).

42. Sastaviti program na jeziku asemblera (da se izvrši pomoću debug.exe) za računanje dva broja: $y_1 = -a - b - 1$ i $y_2 = -2a - 2b - 2$, gdje su a i b dati brojevi. Svi brojevi su veličine po dva bajta. Programski dodijeliti vrijednosti $a = 16$ i $b = 32$ (ovo je u dekadnom).

43. Napisati program na jeziku asemblera (da se izvrši pomoću debug.exe, Intel 8086) za računanje izraza $y_1 = 2a + 2b + 1$ i $y_2 = ab - a - b$, gdje su a i b dati brojevi, recimo $a = 20$ i $b = 24$. Pridružiti promjenljivima a, b, y_1 i y_2 adrese (programer treba da napravi raspored).

44. $y_1 = 2a + 2b + 2$ i $y_2 = ab - 2a - 2b$.

45. $y_1 = 4a + 4b + 1$ i $y_2 = ab - 4a - 4b$.

46. $y_1 = 4a + 4b + 4$ i $y_2 = ab - a - b - 1$.

47. Vježba sa stekom. Upisati u stek redom brojeve (po dva bajta) $x_1 = 11$, $x_2 = 12$, $x_3 = 13$ i $x_4 = 14$ (ovo je u dekadnom sistemu), a onda izračunati (čitanjem sa steka) $y = x_1 + 2x_2 + 3x_3 + 4x_4$.

48. Učitati preko tastature četiri slova (upisati ih u memoriju), prekid INT 21, funkcija AH = 1 ili svejedno AH = 8 (ovo je u heksadekadnom). Zatim štampati u prvom redu učitana slova, a u drugom redu ta ista slova, prekid INT 21, funkcija AH = 2, Carriage return + Line feed = prelazak na početak novog reda. Sastaviti odgovarajući program na jeziku asemblera.

49. Sastaviti program na jeziku asemblera za računanje vrijednosti izraza

$$y = \begin{cases} 2a + 2b. & \text{ako je } c < 0 \\ 8a + 8b, & \text{ako je } c \geq 0 \end{cases}$$

Uputstvo: na primjer, koristiti kombinaciju naredbi CMP i JGE.

50. Program na asembleru za $y = \sum_{k=a}^{b}(k+1)$, pomoću petlje ili pomoću uslovnih skokova, tj. ne koristiti formulu za $1 + \ldots + n$.

U programima na jeziku asemblera (u zadacima) dozvoljeno je da se koriste simboličke labele, kod skokova, kao što je uobičajeno u mnogim asemblerima. Primjer: JMP lab. Ili npr. da umjesto JMP 12C itd. 12C: XOR AX, AX pišemo JMP tamo itd. tamo: XOR AX, AX. Simbolička adresa može da bude "lab" ili "tamo" ili "mjesto" ili neka druga riječ (bilo koja riječ), bilo koja labela.

U nastavku dajemo dva najprostija moguća primjera iz asemblera za procesor Pentium ili slično pod operativnim sistemom Windows. U postavkana zadataka brojevi su prikazani u dekadnom sistemu, a u rješenjima u heksadekadnom. Svi brojevi su veličine po dva bajta.

1. Napisati program na jeziku asemblera za računanje $y_1 = 2a - 5$ i $y_2 = -(b+5)$. Vrijednosti ulaznih podataka $a = 18$ i $b = 14$ treba upisati u registre AX odnosno BX pomoću naredbi MOV. Na kraju, rezultate y_1 i y_2 treba ostaviti u registrima CX odnosno DX.

Slijedi rješenje, a posebno je prikazan sadržaj AX, BX, CX i DX tokom izvršavanja korak po korak.

unosimo naredbe			AX	BX	CX	DX
100	`MOV AX, 12`	AX ← 12	12	0	0	0
103	`MOV BX, E`	BX ← E	12	E	0	0
106	`SHL AX, 1`	AX ← 2 · AX	24	E	0	0
108	`SUB AX, 5`	AX ← AX − 5	1F	E	0	0
10B	`MOV CX, AX`	CX ← AX	1F	E	1F	0
10D	`ADD BX, 5`	BX ← BX + 5	1F	13	1F	0
110	`NEG BX`	BX ← −BX	1F	FFED	1F	0
112	`MOV DX, BX`	DX ← BX	1F	FFED	1F	FFED
114	`INT 20`	stop	1F	FFED	1F	FFED

2. Napisati program na jeziku asemblera za računanje $y_1 = a + b + 1$, $y_2 = a - b - 1$, $y_3 = y_1 \cdot y_2$ i $y_4 = y_1/y_2$. Našim promjenljivim a i b treba dodijeliti vrijednosti $a = 16$ i $b = 12$ pomoću direktiva. Treba ostaviti rezultate $y_1, \ldots, y_4$ na adresama 204, 206, 208 i 20A (hex).

Slijedi rješenje, a posebno je prikazan sadržaj registara AX i BX tokom izvršavanja korak po korak.

unosimo podatke			u memoriji:	
200	`DW 10`	define word (two bytes), $a = 10$	200, 201	10
202	`DW C`	define word (two bytes), $b =$ C	202, 203	C

unosimo naredbe			u procesoru:
100	`MOV AX, [200]`	AX ← c(200)	AX = 10, BX = 0
103	`ADD AX, [202]`	AX ← AX + c(202)	AX = 1C, BX = 0
107	`INC AX`	AX ← AX + 1	AX = 1D, BX = 0
108	`MOV [204], AX`	c(204) ← AX	AX = 1D, BX = 0
10B	`MOV AX, [200]`	AX ← c(200)	AX = 10, BX = 0
10E	`SUB AX, [202]`	AX ← AX − c(202)	AX = 4, BX = 0
112	`DEC AX`	AX ← AX − 1	AX = 3, BX = 0
113	`MOV [206], AX`	c(206) ← AX	AX = 3, BX = 0
116	`MOV AX, [204]`	AX ← c(204)	AX = 1D, BX = 0
119	`MOV BX, [206]`	BX ← c(206)	AX = 1D, BX = 3
11D	`MUL BX`	AX ← AX · BX	AX = 57, BX = 3

11F	`MOV [208], AX`	$c(208) \leftarrow \mathrm{AX}$	$\mathrm{AX} = 57, \mathrm{BX} = 3$
122	`XOR DX, DX`	$\mathrm{DX} \leftarrow 0$	$\mathrm{AX} = 57, \mathrm{BX} = 3$
124	`MOV AX, [204]`	$\mathrm{AX} \leftarrow c(204)$	$\mathrm{AX} = 1\mathrm{D}, \mathrm{BX} = 3$
127	`MOV BX, [206]`	$\mathrm{BX} \leftarrow c(206)$	$\mathrm{AX} = 1\mathrm{D}, \mathrm{BX} = 3$
12B	`DIV BX`	$\mathrm{AX} \leftarrow \mathrm{AX}/\mathrm{BX}$	$\mathrm{AX} = 9, \mathrm{BX} = 3$
12D	`MOV [20A], AX`	$c(20\mathrm{A}) \leftarrow \mathrm{AX}$	$\mathrm{AX} = 9, \mathrm{BX} = 3$
130	`INT 20`	stop	$\mathrm{AX} = 9, \mathrm{BX} = 3$

čitamo pomoću `D 204 20B` (dump) u memoriji:

$y_1 = c(204) = 1\mathrm{D}$ two bytes

$y_2 = c(206) = 3$ two bytes

$y_3 = c(208) = 57$ two bytes

$y_4 = c(20\mathrm{A}) = 9$ two bytes

204, 205	1D
206, 207	3
208, 209	57
20A, 20B	9

Isprobati pomoću nekog programa za propuštanje asemblerskih vježbi.

Na primjer, pomoću programa emu8086. Treba preuzeti program sa adrese www.emu8086.com. Npr. piše: Study computer architecture and assembly language programming. Ili ga preuzeti sa nekog drugog sajta. Dobiće se three months free trial.