7. veljače 2022.

Zavod za elektroniku, mikroelektroniku,
računalne i inteligentne sustave

Arhitektura računala 2

Zimski ispitni rok

1. Na računalo sa 16-bitnom adresnom i 8-bitnom podatkovnom sabirnicom spojeni su memorijski moduli ROM i RAM na sljedeći način:

   - ROM, s priključcima A0-A9 (spojeni na nižih 10 bita adresne sabirnice procesora), D0-D7 (spojeni na podatkovnu sabirnicu procesora), te priključaka CS0-CS3, pri čemu je na CS3 spojeno A15A14 (logički I adresnih linija A15 i A14), na CS2 A13A12 (logički I adresnih linija A13 i A12), na CS1 A11A10 (logički I adresnih linija A11 i A10), a CS0 je trajno priključen na logičku vrijednost 1.
   - RAM, s priključcima A0-A13 (spojeni na nižih 14 bita adresne sabirnice procesora), D0-D7 (spojeni na podatkovnu sabirnicu procesora), priključjka R/W* spojenog (preko odgovarajućih pomoćnih sklopova) na linije R i W procesora, te dva priključaka E i E*, pri čemu je E spojen na adresnu liniju A15, a E* na A14.

   (a) Odredite dijelove adresnog prostora na kojima se javljaju moduli ROM i RAM.

   (b) Napišite program za pojednostavljeni model procesora koji uspoređuje sadržaj memorijske lokacije $A005 sa sadržajem zadnje lokacije ROM-a, te, ako su oni jednaki, uvećava sadržaj lokacije $A006 za 1. Program je smješten počevši od adrese $8000. Na raspolaganju su sljedeće instrukcije: ADDA ($A + M \rightarrow A$, op. kod 88), LDAA ($M \rightarrow A$, op. kod 86), CMPA (uspoređuje A i M i postavlja zastavice, op. kod 81), INC ($M + 1 \rightarrow M$, op. kod 4C), INCA ($A + 1 \rightarrow A$, op. kod 5C), BEQ (ako je Z=1 granaj, ciljna adresa zadana relativno 8-bitnom konstantom, op. kod 27).

   (c) Nacrtajte stanje na vanjskim sabirnicama procesora tijekom izvođenja tog programa.

2. 8-instrukcijskom modelu procesora dodati novu, devetu instrukciju CLR X, koja na memorijsku lokaciju s adresom X upisuje vrijednost 0. Napisati logičke jednadžbe za signale potrebne za fazu izvrši nove instrukcije.

3. Napišite mikrokod instrukcije `CMP_SWAP rj, rk` koja dohvaća vrijednosti s adresa `MEM(rj)` i `MEM(rk)` te u `MEM(rj)` pohranjuje veću od te dvije vrijednosti, dok u `MEM(rk)` pohranjuje manju.

4. Razmatramo procesor s dvorazinskom pomoćnom memorijom L1 i L2. Veličina linije za obje razine je 256B. Veličina L1 je 1kB, dok je veličina L2 20kB. Potrebno je popuniti matricu integera A veličine 64×64 tako da se matrica podijeli na segmente 2×2 u kojoj je svaki element označen indeksom 1-4 (A[0][0]=1, A[0][1]=2, A[1][0]=3, A[1][1]=4...). Predložite kod u c-u koji će ovu matricu popuniti uz minimalan broj promašaja memorije. Koji je omjer pogodaka i promašaja za predloženi kod? Propusnost između procesora i L1 memeorije iznosi 1 2^{30} B/s, a propusnost između L1 i L2 memorije iznsi 1 2^{28} B/s. Ukoliko se na početku izvođenja, cijela matrica A nalazi se u L2, koliko traje 10000 iteracija izvođenja predložene petlje popunjavanja.

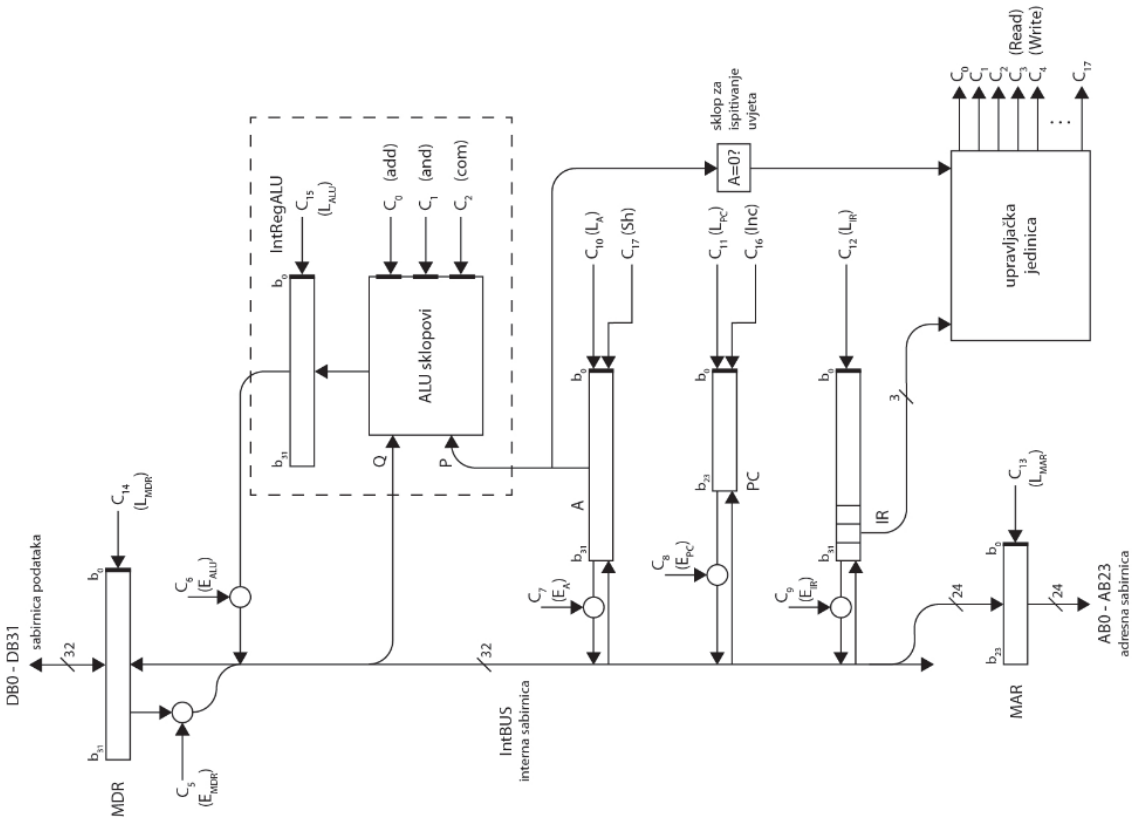
5. Razmatramo funkciju `char *strchr(const char *s, int c);` koja vraća pokazivač na prvo pojavljivanje znaka `c` u znakovnom nizu `s` koji je zaključen nulom. Ako `s` ne sadrži `c` funkcija treba vratiti nulu.

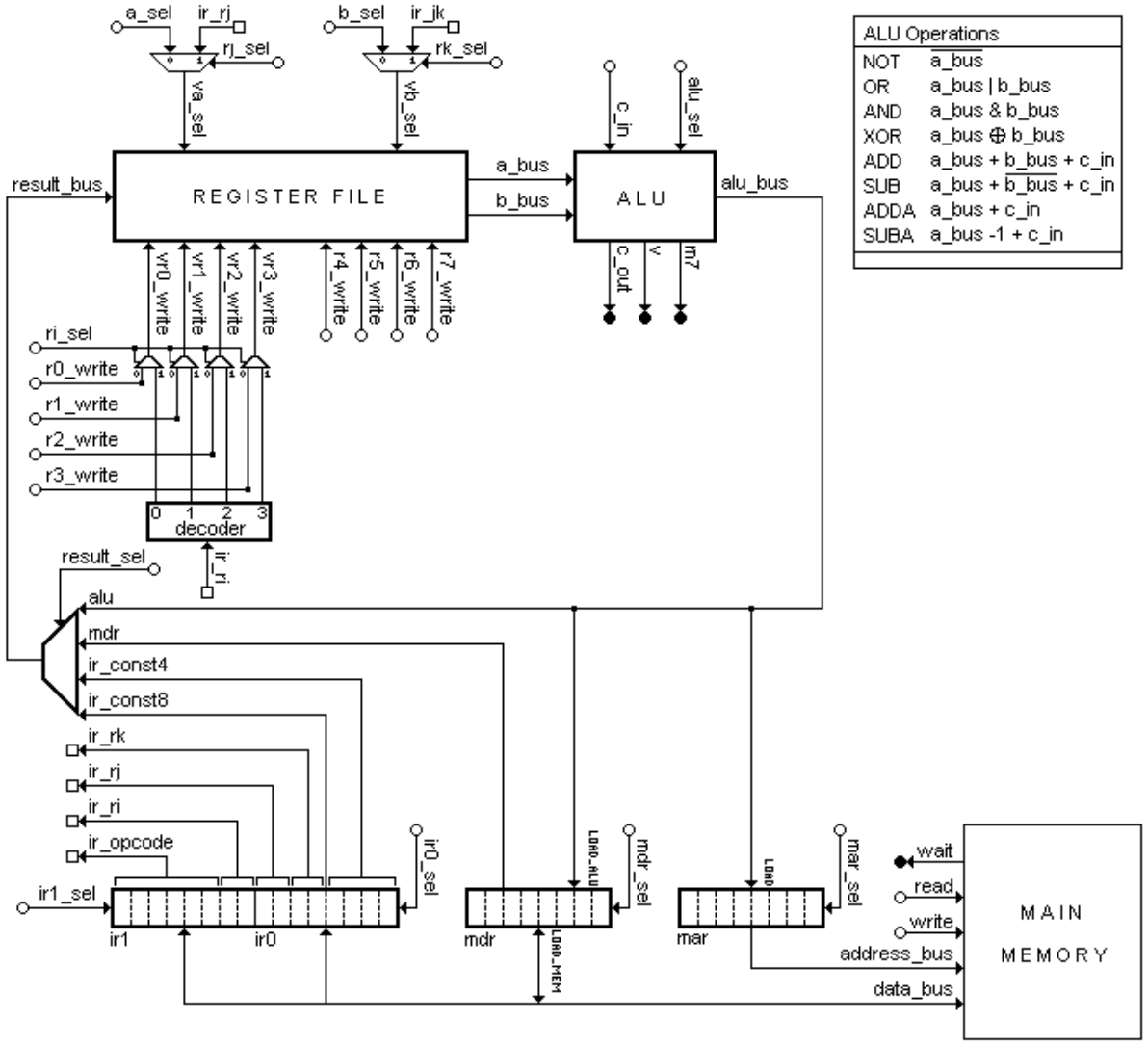
   (a) Napišite implementaciju funkcije u C-u.

(b) Napišite implementaciju funkcije u neoptimiziranom strojnom kodu za procesor arhitekture MIPS pod pretpostavkom zakašnjelog grananja i zakašnjelog učitavanja. Neka se argumenti `s` i `c` prenose preko registara `4` i `5`. Neka se povratna vrijednost prenosi preko registra `2`. Za pohranu privremenih podataka možete koristiti registre `8` do `15`. Registar `0` uvijek sadrži nulu. Priključke svih zakašnjelih instrukcija popunite instrukcijama nop.

(c) Izmijenite strojni kod na način da priključke za kašnjenje popunite korisnim instrukcijama gdje je to moguće tako da se program izvodi čim brže na procesoru s jednostrukim izdavanjem. Neka glavna programska petlja ima samo dvije instrukcije grananja.

(d) Bonus. Prilagodite strojni kod učinkovitom izvođenju na procesoru sa statičkim dvostrukim izdavanjem. Pretpostavite da procesor ima savršeno predviđanje grananja, zbog čega instrukcije grananja ne trebaju priključak za kašnjenje. Prikažite redoslijed izdavanja instrukcija tijekom jednog prolaska kroz petlju.





Memory Interface		
result_sel=	**ALU**	Place the **alu_bus** value on the **result_bus**.
	MDR	Place the mdr value on the **result_bus**.
	IR_CONST4	Place **ir_const4** on the **result_bus**. (**ir_const4** = last 4 bits of ir0)
	IR_CONST8	Place **ir_const8** on the **result_bus**. (**ir_const8** = all 8 bits of ir0)
ir0_sel=	**LOAD**	Load the value on the **memory_bus** into ir0 (Instruction Register 0).
ir1_sel=	**LOAD**	Load the value on the **memory_bus** into ir1 (Instruction Register 1).
mar_sel=	**LOAD**	Load the value on the **alu_bus** into mar (Memory Address Register).
mdr_sel=	**LOAD_ALU**	Load the value on the **alu_bus** into mdr (Memory Data Register).
	LOAD_MEM	Load the value on the **memory_bus** into mdr (Memory Data Register).
read		Place the value at the main memory address mar on the **memory_bus**.
write		Write the value of mdr to the main memory location mar.