Implementacija TCP/IP porta na razvojnom okruženju MSP430 EasyWeb 2

Bojan D. Marić, Radisav R. Ćojbašić

Mentor: docent dr Lazar Saranovac

***Sadržaj* — Tema ovog rada je implementacija TCP/IP protokola na okruženju niske potrošnje, niske cene i malih dimenzija prilagodjenom za internet konekciju. Ovakva realizacija spada u grupu *embedded* sistema, što znači da ima specijalizovanu namenu u odgovarajućim situacijama. Opisana je arhitektura korišćenog razvojnog okruženja. Objašnjena je implementacija TCP/IP protokola. Kao primer provere ispravnosti rada realizovana je dinamička web stranica.**

***Ključne reči* — API set funkcija, Dinamički HTTP server, *Ethernet* kontroler, Komunikacioni protokoli, LAN, Mikrokontroler.**

I. Uvod

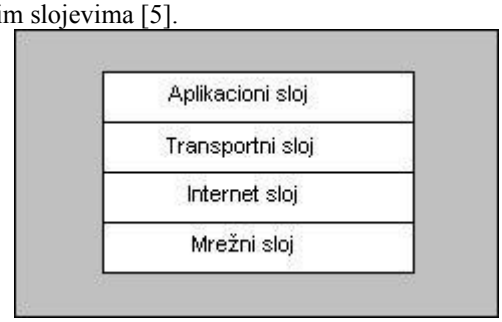
Upoteba *ethernet*-a u računarskim mrežama je sve masovnija zbog niske cene za izradu aplikacija prilagodjenih za mrežnu komunikaciju. Računarski komunikacioni sistemi igraju važnu ulogu u našim životima, ne samo u domenu personalnih računara, nego i u domenu malih lokalnih mreža- LAN. Veliki broj današnjih uređaja za svakodnevnu upotrebu se ne može zamisliti bez ugrađenog jednog ili više procesora. U takvim sistemima se razmena podataka obavlja po nekom komunikacionom protoklu. U daljem tekstu je prikazana implementacija TCP/IP protokola primenom *ethernet* kontrolera. Zapravo, korišćen je razvojni hardver sa mikrokontrolerom i kontrolerom *ethernet*-a, prilagodjen za umrežavanje i internet konekciju poznat pod nazivom EasyWeb 2 [1]. Mikrokontroler na toj ploči je iz serije MSP430 firme *Texas Instruments* [2]. Njegova osnovna osobina je izuzetno niska potrošnja, bogat instrukcijski set kao i raznovrsnost integrisanih periferija. Kao provera ispravnosti funkcionisanja implementacije, podignuta je dinamička web stranica. Na toj stranici su prikazani rezultati merenja koja se obavljaju pomoću mikrokontrolera (tačnije vizuelni prikaz rezultata A/D konverzije napona na odgovarajućem pinu mikrokontrolera). Postoji i mogućnost kontrole rada mikrokontrolera preko kontrolnog dugmeta na podignutoj web stranici (tačnije ispis poruke na LCD displeju ako se klikne ne dugme). Takodje je prikazana i dijagnostika koja nam prikazuje sve IP adrese uredjaja koje su trenutno u LAN-u. Detalje vezane za mikrokontroler kao i kontroler *ethernet*-a možete naći u diplomskom radu Radisava Ćojbašića [3]. Detaljan opis softvera pomoću koga se implementira TCP/IP protokol na EasyWeb2 razvojnom okruženju kao i osnovi mehanizma TCP protokola možete naći u diplomskom radu Bojana Marića [4].

. Bojan D. Marić, Elektrotehnički fakultet u Beogradu, Srbija (telefon: +381-65-6223059; e-mail: bozzzan@gmail.com)

Radisav R. Ćojbašić, Elektrotehnički fakultet u Beogradu,Srbija (telefon: +381-63-259797; e-mail: cojbasic@yahoo.com).

II. Kratak opis komunikacionih protokola

Protokol stekovi se opisuju pomoću slojevitog modela kao što je prikazano na Sl. 1. Svaki od slojeva obezbedjuje odgovarajuće funkcije protokol stekovima u gornjim slojevima, kao i odgovarajuće servise protokol stekovima u donjim slojevima [5].



Sl. 1. Slojeviti model protokol stekova.

A. TCP/IP protokol

TCP protokol pripada transportnom sloju i pomoću njega se ostvaruje pouzdana komunikacija (praćenje prenosa, retransmisija, itd.). Komunikacija se ostvaruje samo u slučaju da su i server i klijent istovremeno konektovani (zato je ovaj protokol pouzdan).

IP protokol pripada internet sloju i koristi se najčešće u internet mrežama. IP protokol obezbedjuje mehanizam za prenos podataka od izvora ka odredištu. Učesnici u komunikaciji su idetifikovani pomoću jedinstvenih IP adresa. Medjutim IP protokol ne obezbedjuje pouzdan vid komunikacije, pa se zato za komunikaciju koriste protokoli iz višeg sloja kao što je TCP protokol. IP protokol ne obraća pažnju da li je klijent konektovan na mrežu ili da li je podatak koji se šalje/prima oštećen prilikom slanja/prijema.

B. ARP protokol

ARP protokol se najčešće koristi u mrežama koje koriste *ethernet*. Pomoću ARP protokola pronalazimo fizičku (MAC) adresu uredjaja sa kojim komuniciramo. Pronalaženje fizičke adrese se obavlja slanjem *broadcast* paketa ka svim uredjajima u mreži i prijemom odgovarajućih *acknowledgement* paketa i uspostavljanja komunikacije na višem (transportnom) sloju.

C. ICMP protokol

ICMP protokolom se obezbedjuje mehanizam za obaveštavanje ako se nešto uspešno ili neuspešno odradilo

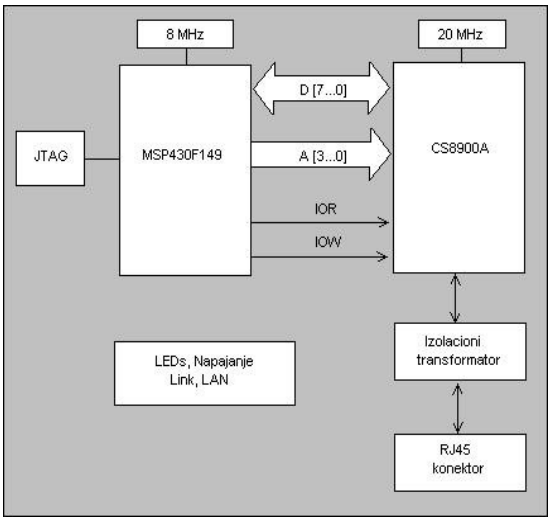
(recimo kad odredjeni paket podataka nije poslat ili kad nije prepoznata fizička adresa uredjaja, itd.).

D. HTTP protokol

HTTP protokol pripada aplikacionom sloju. HTTP protokol koristi klijent-server odnos koji je baziran na transportnom sloju kao što je TCP. Danas se najviše koristi za transfer HTML dokumenata izmedju klijenta i servera. Radi na principu zahteva i odgovora.

III. Opis hardvera

Dve glavne komponente EasyWeb 2 razvojnog okruženja su mikrokontroler MSP430F149 firme *Texas Instruments* i kontroler *ethernet*-a CS8900A, čiji je proizvodjač *Crystal*^{TM} *Semiconductor Corporation* [2], [6]. Mikrokontroler MSP430F149 je poznat kao *ultra low-power* komponenta koja poseduje 60KB fleš memorije i 2KB RAM memorije. Zbog toga je pogodan za smeštanje i transfer koda web stranica. Takodje sadrži šest 8-bitnih I/O bidirekcionalnih portova, koji služe u povezivanju mikrokontrolera i kontrolera *ethernet*-a. Kontroler *ethernet*-a CS8900A je jeftina komponenta optimizovana za standardne personalne računare. CS8900A je veoma pogodna komponenta za ovu aplikaciju, jer je komponenta sa visokim stepenom integracije i veoma je lako upravljati interfejsom za magistralu. Mnogi kontroleri *ethernet*-a na tržištu imaju PCI interfejs magistralu, dok CS8900A može direktno da se poveže na mikrokontroler. CS8900A se napaja sa 3V kao i MSP430F149, što je još jedna prednost pri povezivanju ove dve komponente. Na Sl. 2 je prikazana uprošćena šema EasyWeb 2 razvojnog okruženja [3].



Sl. 2. Blok šema razvojnog hardvera.

Najinteresantnija stvar na hardveru EasyWeb2 je povezivanje mikrokontrolera i kontrolera *ethernet*-a. CS8900A može da radi u tri režima rada: u I/O prostoru, memorijskom prostoru i kao DMA *slave.* Svaki od ovih modova rada ima svoje prednosti i mane. Za ovu aplikaciju je režim rada u I/O prostoru najpodesniji. Ovo je podrazumevani-*default* mod rada CS8900A i uvek je dozvoljen, čim se podigne napajanje. Najvažnija stvar je što koristimo 8-bitnu magistralu za podatke. Ova magistrala za podatke je direktno povezana na 8-bitni I/O port broj 5 mikrokontrolera MSP430F149. Pristup CS8900A kada je u I/O režimu rada se vrši preko osam 16-bitnih I/O portova koji su mapirani u šesnaest 8-bitnih registara. Da bismo pristupili ovim registrima potrebna je 4-bitna adresna magistrala. Takodje, postoje još dve kontrolna linije IOW i IOR. Ovi signali su aktivni u logičkoj nuli i pokazuju da li se radi o procesu upisa ili čitanja u registre CS8900A. Na kraju možemo videti koliko je jednostavan interfejs izmedju CS8900A i MSP430F149 i da se on sastoji od 14 električnih linija. Posle postavljanja validne adrese na adresnu magistralu i spuštanja odgovarajućeg kontrolnog signala (IOW ili IOR) na nulu, transfer podatka preko magistrale za podatke može početi. Razlog zbog čega je jednostavno povezivanje CS8900A i MSP430F149 leži u tome što unutar CS8900A postoji *10Base-T transceiver* koji ima komplentnu analognu i digitalnu elektroniku potrebnu za implementiranje LAN interfejsa upotrebom jednostavnog izolacionog transformatora. Zbog toga nije potrebno ništa dodavati spolja, sem spoljnih kondenzatora za oscilator. To u mnogome olakšava pisanje drajvera za CS8900A, jer mi samo treba da pošaljemo podatak ka CS8900A, a unutar njega se vrši filtriranje i pakovanje po odgovarajućem protokolu i slanje u LAN. Slično je i za prijem podatka. Klasični RJ45 kabl je upotrebljen za povezivanje u LAN. Tipična brzina prenosa za CS8900A je 10Mbps. EasyWeb 2 hardversko okruženje sadrži i JTAG interfejs koji služi za direktno povezivanje *Flash Emulation Tool*-a i omogućavanja programiranje i debagovanje aplikacije.

IV. Opis softvera

A.Osnovne API funkcije za implementaciju TCP steka

U ovom poglavlju će biti objašnjene osnovne API funkcije (gotove funkcije koje se samo pozivaju) korišćene za implementaciju TCP/IP steka [4], [5]. To su:

- **void TCPLowLevelInit(void)-** Ova funkcija vrši osnovnu inicijalizaciju *ethernet* kontrolera i numeričkih promenljivih. Takodje , posle poziva ove funkcije se konfigurišu portovi i tajmer A mikrokontrolera. Pre svakog transfera podataka preko TCP/IP steka je neophodno pozvati TCPLowLevelInit.
- **void TCPPassiveOpen(void)-** Pozivom ove funkcije implementirani stek se prebacuje u serverski mod rada i sada osluškuje dolazeću konekciju. Posle ovoga fleg SOCK_ACTIVE se setuje i pokazuje da je sada stek zauzet za eventualno novu konekciju. Pre poziva ove funkcije mora biti postavljen lokalni TCP port kao i lokalna IP adresa. Sada ako klijent uspešno ostvari konekciju sa serverom, fleg SOCK_CONNECTED se setuje u registru *SocketStatus* [4].
- **void TCPActiveOpen(void)-** Ova funkcija pokušava da uspostavi konekciju sa udaljenim serverom nakon odradjene pasivne konekcije. Fleg SOCK_ACTIVE je setovan i sada se šalje ARP zahtev za pronalaženjem fizičke (MAC) adrese uredjaja. IP adrese kao i lokalni i udaljeni TCP portovi moraju biti podešeni pre poziva ove funkcije. Posle otvaranja konekcije, sinhronizacija TCP mašine stanja počinje kao

što je objašnjeno u [4] i sada će se nalaziti u stanju ESTABLISHED. SOCK_CONNECTED je setovan u registru *SocketStatus* i transfer podataka po TCP/IP protokolu može da počne pozivom odgovarajućih API funkcija. Ako se desi neka greška pri uspostavljanju konekcije, konekcija se prekida i odgovarajući *error code* se smešta u *SocketStatus* registar.

- **void TCPClose(void)-** Ova funkcija zatvara otvorenu konekciju. Pre zatvarajna konekcije stek obezbedjuje da i poslednji paket podataka koji se nalazi u baferu bude poslat i potvrdjen od strane prijema. Posle zatvaranja konekcije, korisnik može da rekonfiguriše IP adrese, brojeve portova i otvori novu konekciju.
- **void TCPReleaseRxBuffer(void)-** Posle čitanja prijemnog bafera, pozivom ove funkcije obaveštavamo stek da sadržaj bafera više nije bitan i da može biti upisana nova vrednost TCP podatka. Ova funkcija takodje briše SOCK_TX_BUF_RELEASED fleg koji označava prisutnost novog podatka u baferu.
- **void TCPTransmitTxBuffer(void)-** Pozivom ove funkcije je moguće slanje podataka preko pethodno otvorene konekcije. Prvo, aplikacija mora proveriti da li se može upisati podatak u transmisioni bafer. Ovo možemo uraditi proverom SOCK_TX_BUF_RELEASED flega u *SocketStatus* registru. Ako je pomenuti fleg setovan aplikacija može upisati maksimalno MAX_TCP_TX_DATA_SIZE bajtova u transmisioni bafer počevši od adrese TCP_TX_BUF. Broj bajtova mora biti upisan u TCPTxDataCount registar. Konačno, pozivom funkcije TCPTransmitTxBuffer() počinje slanje podataka.

Struktura *SocketStatus* registra je detaljno prikazana i objašnjena u [4]. Najvažnija funkcija TCP/IP steka je *DoNetworkStuff* () [4]. Ova funkcija bi trebala periodično da se poziva od strane korisničkog programa. Ovde se obavlja obradjivanje dogadja vezanih za TCP protokol. Takodje, funkcija *DoNetworkStuff* () vrši poliranje raznih flegova vezanih za mikrokontroler kao i kontroler *ethernet*-a. U skladu sa tim flegovima, funkcija radi odgovarajuće poslove (da li je stigao odgovarajući segment podatka, da li je isteklo vreme potrebno za obradu odgovarajućeg segmenta, tj. da li je potrebno izvršiti retransmisiju, da li je stigao zahtev za slanje od odgovarajućih bafera i postavljanje statusnih bita TCP mašine stanja u odgovarajuće stanje). Specijalan set poslova koje obavlja ova funkcija su poslovi nakon korisničkih zahteva. Ti poslovi se izvršavaju pozivanjem funkcija implementiranog steka direktno od strane aplikacije.

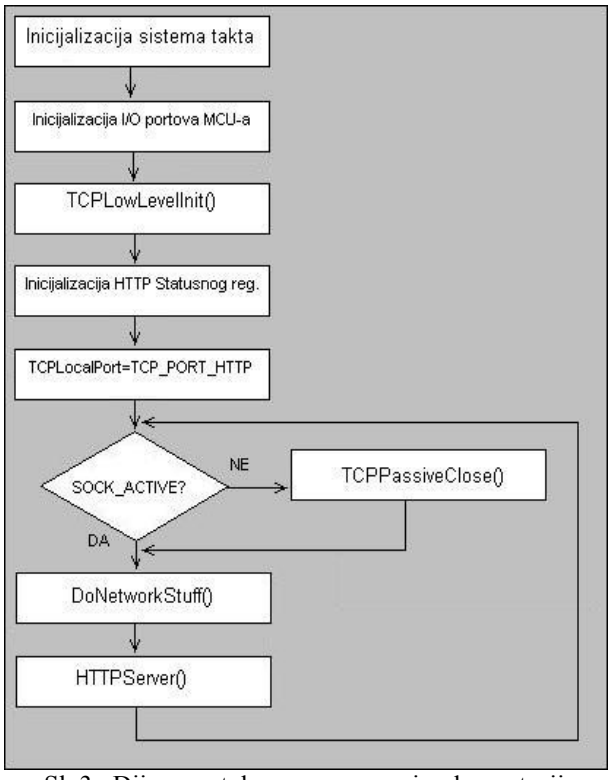
B.Ethetnet modul

U okviru drajvera za *ethernet* kontroler se nalaze funkcije koje koriste set API funkcija za transfer podataka, prijem i slanje frejmova i sve ono što je potrebno da bi se ostvario prenos po TCP/IP protokolu. Pre nego što se počne sa radom, potrebno je izvršiti podešavanja vezana za mrežni interfejs. To obuhvata i podešavanje MAC adrese mrežnog interfejsa. Ova adresa je 48-bitna i definisana je pomoću šest simboličkih konstanti MYMAC_1,...,MYMAC_6. Korisnik može da menja ovu adresu, ali mora biti siguran da je ona jedinstvena unutar mreže. Adresu FFFF FFFF FFFF FFFF nije dozvoljeno koristiti kao MAC adresu, jer je rezervisana za *broadcast* adresu. U [4] su prikazane funkcije za drajvovanje *ethernet* kontrolera. Prvo se radi inicijalizacija sistema pozivom odgovarajuće funkcije. Posle toga korisnički program može da radi svoj posao sve do trenutka kada treba izvršiti slanje ili prijem podataka korišćenjem *ethernet* modula. Sada možemo da vršimo slanje i prijem podataka. Što se tiče prijema podataka, proverava se prijemni bafer i kopira se primljeni frejm u odgovarajući registar kontrolera *ethernet*-a, a zatim u memoriju mikrokontrolera. U slučaju slanja frejma prvo se pošalje zahtev za slanje frejma odgovarajuće veličine. Posle toga se proverava da li je prethodno zahtevani prostor za transmisioni bafer u kontroleru *ethernet*-a dostupan. Ako jeste, još jednom se šalje zahtev za slanje koji će biti prihvaćen i slanje se izvršava.

C.Dinamički HTTP server

Kao primer provere rada implementiranog TCP/IP steka prikazaćemo realizaciju dinamičkog HTTP servera. HTML web stranica servera je smeštena u fleš memoriji mikrokontrolera. Ovaj modul očekuje dolazeću konekciju, vrši transfer web stranice, zatvara konekciju i čeka druge klijente da se konektuju. Na web stranici su prikazana dva bara grafa, koji prikazuju rezultat A/D konverzije na portu 6.7 i temperaturu mikrprocesora. Takodje je prikazana i dijagnostika TCP/IP-a, jer su prikazane IP adrese uredjaja koji se nalaze u LAN-u (klijenti i server). Pored toga implementirana je kontrola hardvera preko podignute web stranice tako što postoji *submit button*-dugme na čiji pritisak se vrši odgovarajuća akcija na hardveru (u našem slučaju ispis poruke na LCD displej). HTTPServer() je funkcija koja pomoću HTML koda implementira dinamički HTTP server. Pre početka rada programa moraju se podesiti IP adrese servera i udaljenog klijenta kao i broj lokalnog TCP porta (za HTTP server je to 80). Server sada čeka da se klijent konektuje na mrežu. Posle prvog poziva funkcije HTTPServer() fleg HTTP_SEND_PAGE u registru HTTPStatus se briše. Web server proverava dolazeći TCP podatak i odbacuje ga. Primljeni podatak predstavlja GET zahtev korišćenog internet *browser*-a. Server počinje sa radom upravo kada se klijent konektuje i pošalje Web stranu smeštenu u fleš memoriji. Posle provere statusa transmisionog bafera pokazivač na WebSide[] je inicijalizovan i ukupan broj bajtova za slanje je smešten u HTTPBytesToSend. Prilikom prvog poziva funkcije HTTPServer(), odgovarajući HTTP *header* je direktno prenesen pre web stranice. To je obaveštenje klijentu da je zahtev uspešno dekodovan i koja vrsta resursa se prenosi (HTML). To je smešteno u konstanti GetResponse[] u fajlu easyweb.h. Posle toga web strana se šalje u paketima veličine MAX_TCP_TX_DATA_SIZE. Opisano uspešno slanje cele web strane se okončava pozivom funkcije TCPClose(). Na Sl. 3 je prikazan dijagram toka programa za implementaciju dinamičke web stranice. Web stranica se osvežava na svakih 5s. Dovoljno je otvoriti Internet

Explorer i da korisnik u polju za adresu sajta otkuca *http://* i IP adresu lokalnog hosta (10.10.1.30) i stranica će biti otvorena.

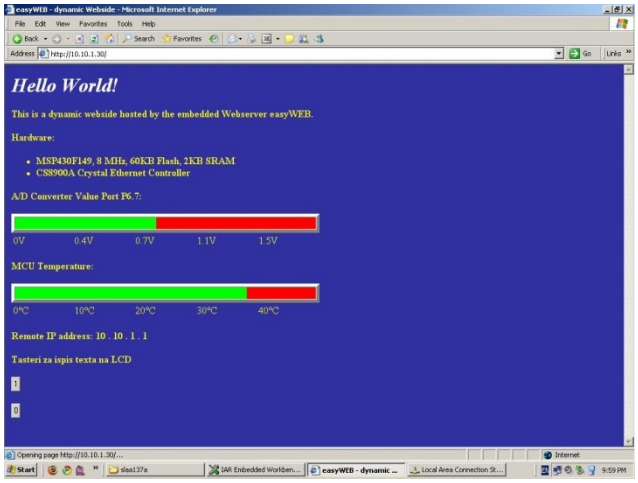


Sl. 3. Dijagram toka programa za implementaciju dimamičke web stranice.

Na Sl. 4 je prikazan izgled podignute web stranice u Internet Explorer-u.

V. Zaključak

U radu je opisana implementacija nekoliko komuniacionih protokola na *embedded* sistemu. Navedeni su osnovni razlozi izbora elektronskih komponenti (mikrokontrolera i kontrolera *ethernet*-a) i njihovo povezivanje u električno kolo. Ovaj primer pokazuje da je moguće rešiti veoma kompleksne probleme pomoću gotovih sistema, niske cene, kao što su mikrokontroleri. Pitanje je samo brzina rada i performanse koje nam trebaju da bi uredjaj pouzdano radio. Mogućnosti proširenja sistema su velike. Pre svega moguće je upotrebiti neki drugi mikrokontroler (naravno pogodan za ovakvu realizaciju) ili neki drugi odgovarajući kontroler *ethernet*-a, a da koncept sistema ostane približno isti. Prednosti ovakvog sistema su mala potrošnja i kontrola sistema na daljinu. Uredjaj je pogodan za baterijsko napajanje i za instalaciju na mestima gde je električna energija nepristupačna. Ovakva realizacija ne zahteva virtuelnu instrumentaciju, tj. poseban softver koji je pisan samo za potrebe tog mernog uredjaja. Takav inteligentni senzor je kompatibilan sa svakim računarom koji ima internet pretraživač. Ove osobine pružaju mogućnost za realizaciju mnogih aplikacija koje funkcionišu na sličan način. Neke od nih



Sl. 4. Izgled podignute web stranice u Internet Explorer-u.

su: čitači kartica, sigurnosni sistemi, automatizovano merenje fizičkih veličina, itd.

VI. Literatura

[1] http://www.olimex.com/dev/index.html

[2] Texas Instruments, "*MSP430x14xx Family User's Guide*",Vol3.,February2005

[3] R. Ćojbašić, "*Hardverksa podrška TCP/IP portu realizovanom na razvojnom okruženju MSP430*", diplomski rad, Elektrotehnički fakultet, Beograd

[4] B. Marić, "*Programska podrška TCP/IP portu realizovanom na razvojnom okruženju MSP430*", diplomski rad, Elektrotehnički fakultet, Beograd

[5] Washburn, K., Evans, J. , "*TCP/IP Running a Successful Network*", Addison Wesley, 1996.

[6] Cirrus™ Logic, "*CS8900A Product Data Sheet*", Inc., 1999.

Abstract

In this paper is presented implementation of the TCP/IP protocol on a low-power, low-cost and small dimensions development board that is adapted for internet connectivity. This realization is embedded system, and it's means it has a special assignment in a right situation. In this paper is explained architecture of the development board. Also the software for implementation of the TCP/IP is explained. A Dinamic web page is implemented to show a verification of the propriety for realised TCP/IP stack.

IMPLEMENTATION OF THE TCP/IP STACK ON DEVELOPMENT BOARD MSP430 EASYWEB 2

Bojan D. Marić, Radisav R. Ćojbašić