Jedno rešenje alata za analizu automobilskih magistrala korišćenjem osciloskopa

Filip Jašić, Dejan Bokan i Zoran Marčeta

***Sadržaj* — Jedan od ciljeva rada je bio da se objasni implementirano snimanje, obrada i dekodovanje signala protokola koji se koristi u automobilskoj industriji – *CAN*-a, upotrebom osciloskopa. Pored pomenutog, potrebno je bilo ostvariti automatizaciju rada osciloskopa. Automatizovan proces dekodovanja protokola uveliko olakšava dijagnostiku automobilskih mreža time što korisniku daje detaljan prikaz magistrala celokupnog sistema.**

Ključne reči — automobilske magistrale, osciloskop, obrada signala, dekodovanje.

I. Uvod

Glavni pokretač razvoja mrežnih tehnologija u automobilima su bili napreci ostvareni u elektronskoj industriji. Zbog striktne kontrole štetnih gasova, morao se napraviti mehanizam za veći stepen kontrole samog automobila. Ovo se postiglo ugrađivanjem računara koji ne samo da su poboljšali upravljanje emisijom štetnih gasova, već su povećali i performanse automobila, sigurnost, komfor, a doprineli su i smanjenju troškova proizvodnje. Tokom godina se broj elektronskih modula u automobilu povećavao.

Elektronska upravljačka jedinica (engl. *ECU*) predstavlja ugrađeni sistem koji upravlja jednim ili sa više elektronskih modula. U modernim vozilima može da se nađe i do 80 elektronskih upravljačkih jedinica. *ECU* dobija podatke od senzora koji se konvertuju u određenu jedinicu, gde se tako obrađeni podaci dalje šalju aktuatorima koji izvršavaju određene funkcije. Ponekad je potrebno da ovi moduli međusobno komuniciraju. Kako se vremenom povećavao broj ovih komponenti, došlo je do velike kompleksnosti u ožičavanju pojedinih modula. Štaviše, ožičavanja modula automobila su se razlikovala od modela do modela, što je dodatno povećavalo troškove proizvodnje. Odgovor automobilske industrije na prethodno opisane probleme je bio da se stvori centralna mreža u vozilu, a svaki modul, koji predstavlja čvor na toj mreži, kontroliše specifičnu komponentu i komunicira sa drugim modulima po potrebi korišćenjem standardnih protokola.

Ovaj rad je delimično finansiran od strane Ministarstva za prosvetu, nauku i tehnološki razvoj Republike Srbije, na projektu broj: III_044009_1.

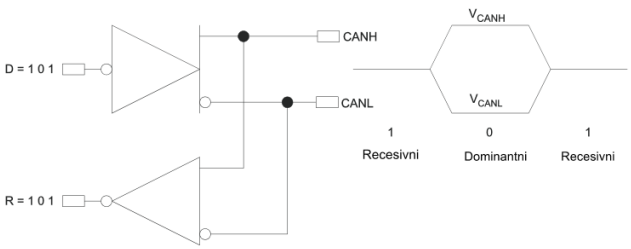
Filip Jašić, Istraživačko -razvojni Institut RT-RK, Novi Sad, Srbija (e-mail: filip.jasic@rt-rk.com).
Dejan Bokan, Istraživačko-razvojni Institut RT-RK, Novi Sad, Srbija (e-mail: dejan.bokan@rt-rk.com).
Zoran Marčeta, Istraživačko-razvojni Institut RT-RK, Novi Sad, Srbija (e-mail: zoran.marceta@rt-rk.com).

Jedan od zadataka ovog rada jeste realizacija dekodovanja signala protokola čija je upotreba česta u automobilskoj industriji, čime bi se olakšala analiza celokupnog sistema. Čestu primenu u dijagnostici automobilskih mreža ima osciloskop. Jedan od ciljeva ovog rada jeste i automatizacija podešavanja osciloskopa, a dekodovanjem signala protokola se dobija čitljiviji prikaz eventualnih grešaka ili problema na mreži, što dodatno ubrzava proces dijagnostike.

II. Protokol

CAN (engl. *Controller Area Network*) protokol je najzastupljeniji protokol koji se koristi u automobilskoj industriji. Verzija specifikacije se sastoji iz dva dela:

- A (pokriven *ISO* 11519 standardom [1]) – ovaj deo specifikacije opisuje standardni format sa 11-bitnim identifikatorom.
- B (pokriven *ISO* 11898 standardom) – ovaj deo specifikacije opisuje prošireni format sa 29-bitnim identifikatorom.



Sl. 1 Razlike napona na linijama i povezanost sa reprezentacijom bita u vidu 1 ili 0.

CAN mreža poseduje mogućnost da ima više rukovodioca. Sve elektronske upravljačke jedinice se međusobno povezuju na magistralu sa dve linije (jedna se zove *CAN-H*, druga *CAN-L*). Dominantni bit predstavlja logičku nulu. Predstavljanje logičke nule kao dominantnog bita se vrši da bi elektronske upravljačke jedinice sa najnižim identifikacionim brojem imale najveći prioritet. Recesivni bit predstavlja logičku jedinicu. Logička stanja se procenjuju na osnovu razlike napona (Sl. 1):

- Za brzu *CAN* magistralu (*ISO* 11898-2) [2]:
  - Recesivni bit predstavlja razliku od 0 *V* (u teoriji) između *CAN-H* i *CAN-L* linije; u praksi se ta razlika kreće od 0.5 *V*. Napon na *CAN-H* žici se kreće ka 5 *V*, dok se na *CAN-L* žici kreće ka 0 *V*.

978-1-7281-4790-1/19/$31.00 ©2019 IEEE

- Dominantni bit predstavlja razliku od oko 2 *V* (mora bitu u rasponu između 1.5 *V* i 3.5 *V*) između *CAN-H* i *CAN-L* linije. Napon na *CAN-H* žici se kreće ka 5 *V* dok napon na *CAN-L* žici se kreće ka 0 *V*.

- Za sporu *CAN* magistralu (ISO 11898-3):
  - Recesivni bit predstavlja razliku od bar 0.6 *V* između *CAN-H* i *CAN-L* linije. Napon na *CAN-H* žici se kreće ka 0 *V*, dok napon na žici *CAN-L* ide ka 5 *V*.
  - Dominantni bit predstavlja razliku od bar 2.3 *V* između *CAN-H* i *CAN-L* linije. Napon na *CAN-H* žici se kreće ka 5 *V*, dok napon linije *CAN-L* ide ka 0 *V*.

Prenos poruka je kontrolisan sa četiri tipa okvira:

- Okvir za podatke – nosi podatke od predajnika do prijemnika. Okvir za podatke se sastoji od sedam polja (razlike između *CAN 2.0 A* i *CAN 2.0 B* su po potrebi posebno navedeni, dok se podrazumeva da ostala polja imaju istu strukturu):
  - Početak okvira označava početak polja za okvir za podatke ili okvira za daljinsko upravljanje. Sastoji se od jednog dominantnog bita.
  - Polje za arbitražu. Ovo polje se sastoji od identifikatora i *RTR* bita u slučaju *CAN 2.0 A*, pri čemu identifikator se sastoji od 11 bitova; ovi biti se gledaju u *MSB* redosledu pri čemu svih 7 najznačajnih bitova (od *ID-10* do *ID-4*) ne smeju biti recesivni. *RTR* bit mora da bude dominantan kada se šalje okvir za podatke, inače, kada se šalje okvir za daljinsko upravljanje ovaj je bit recesivan. U slučaju *CAN 2.0 B* identifikator se sastoji iz dva dela: identifikator A (11 bitova) i identifikator B (18 bitova). Između ova dva dela produženog identifikatora se nalazi *SRR* bit (1 recesivan bit) i *IDE* bit (1 recesivan bit).
  - Kontrolno polje. Ovo polje se takođe sastoji od 6 bitova, pri čemu prva 2 bita rezervisana. Nakon njih u okviru se nalaze 4 bita koji označavaju dužinu poslatih podataka.
  - Polje za podatke se sastoji od 0 do 64 bitova. Podaci se šalju u *MSB* redosledu.
  - Polje za cikličnu proveru redudanse (engl. *CRC field*) se sastoji od 15 bitova koji predstavljaju izračunatu cikličnu proveru redudanse i delimitera koji zauzima 1 bit, pri čemu je taj bit recesivan.
  - Polje za potvrdu o prijemu se sastoji od *ACK* vremenskog intervala i *ACK* delimitera gde svaki od polja zauzima po 1 bit i pri čemu pošiljaoc postavlja recesivni bit u *ACK* vremenskom intervalu, dok primaoc poruke može da naredi slanje dominantnog bita. *ACK* delimiter „razdvaja“ ovo polje sa krajem okvira i ono sadrži 1 recesivan bit.
  - Kraj okvira se sastoji od 7 recesivnih bitova.
- Okvir za daljinsko upravljanje se šalje na magistralu kao zahtev za okvirom za podatke sa određenim identifikatorom.
- Okvir za greške se šalje od bilo kog čvora uređaja koji detektuje grešku na magistrali.
- Okvir za preklapanje se koristi da omogući kašnjenje između dva poslata okvira za podatke ili okvira za greške.

Kada god je magistrala slobodna, bilo koja elektronska upravljačka jedinica može da počne sa slanjem poruke. U slučaju da dve ili više upravljačkih jedinica počne sa slanjem poruka u isto vreme nastaje konflikt koji se rešava mehanizmom arbitraže za pojedinačne bite korišćenjem identifikatora.

Tokom arbitraže, svaki pošiljalac proverava na kom nivou se nalazi bit identifikatora koji je on poslao poredeći ga naspram nivoa bita identifikatora koji se nalazi na magistrali. Ako su ovi biti na istim nivoima jednaki, pošiljalac može da nastavi sa slanjem. Kada je recesivni bit na istom nivou poslat i dominantni bit na istom nivou se nalazi na magistrali, tada je pošiljalac „izgubio“ arbitražu i mora da se povuče, pri čemu se prekida prenos preostalih bita u okviru. Drugim rečima, pobeđuju uređaji sa najmanjim identifikatorom. Identifikator se šalje u *MSB* (engl. *Most Significant Bit*) formatu, da bi se brže odigrala arbitraža i da bi medijum bio efikasniji za slanje.

Ubacivanje bita se vrši na sledeći način: nakon 5 identičnih bitova se ubacuje suprotan bit; ovo poboljšava usklađivanje, a dodatni biti ne menjaju podatke pošto je ubacivanje urađeno hardverski. Problem sa ubacivanjem bita je pojava takozvanih „lavinskih“ bitova (npr. 5 recesivnih bitova koje prati 5 dominantnih bitova) koji mogu da zbune mehanizam za prepoznavanje ubačenih bita.

Statistička učestanost grešaka *CAN* protokola zavisi od ukupnog broja uređaja, fizičkog ožičenja i rasporeda, kao i spoljašnjih elektromagnetnih smetnji. Tok prepoznavanja grešaka između pošiljaoca i primaoca je nezavisan od filtriranja i maskiranja na prijemu i sastoji se iz pet zasebnih mehanizama:

1. Nadgledanje pojedinačnih bita – prepoznavanje lokalnih i globalnih grešaka kod pošiljaoca; upoređuje se stanje na magistrali sa poslatim bitom; ne primenjuje se na polje za arbitražu.
2. Provera strukture – tipična polja su *CRC* delimiter, *ACK* delimiter i *EOF* i čine ih uvek recesivni biti.
3. Provera kodovanja (umetanje bita) – prijemnik proverava tok bita, na svakih 5 uzastopnih istovetnih bitova mora da dođe ubačena promena; uključuje sve bite od *SOF* do kraja *CRC*.
4. Provera potvrde je obaveza pošiljaoca – potrebno je bar da pristigne jedna potvrda pa da pošiljalac podesi recesivni bit; takođe, potrebno je da prijem obori sa dominantnim.
5. Ciklična provera redudanse – pošiljalac proračunava pre slanja, prijemnik proverava podudaranje pri prijemu. Ciklična provera redudanse se računa na sledeći način: polinom, čiji su koeficijenti predstavljeni kombinacijom vrednosti prethodnih polja (polje za početak okvira, polje za arbitražu, kontrolno polje, polje za

podatke), pri čemu je prethodno urađeno izbacivanje ubačenih bita, a koeficijenti 15 najnižih bitova čine nule. Ovaj polinom se potom deli (koeficijenti su izračunati modulom dvojke) sa generator-polinomom:

$x15 + x14 + x10 + x8 + x7 + x4 + x3 + 1$ [1]

Ostatak ovog deljenja predstavlja 15-bitnu vrednost *CRC* sekvence koja se šalje na magistralu kao dodatak originalne poruke. Da bi se primenila ova funkcija, 15-bitni koristi se pomerački registar.

III. Implementacija

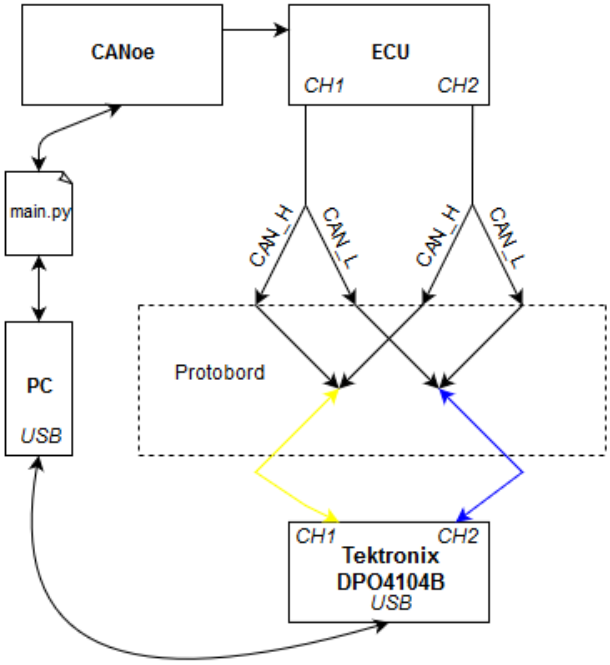
Za generisanje signala za *CAN* protokol korišćena je elektronska upravljačka jedinica i alat *CANoe*. *CANoe* je napredni alat kompanije Vektor Informatik [3] za razvoj, testiranje i analizu kako pojedinačnih elektronskih upravljačkih jedinica, tako i kompletne mreže.

U *CANoe* je prvo kreirana odgovarajuća konfiguracija i sačuvana u konfiguracionoj datoteci sa nastavkom *.cfg*. Potom je u *CANoe Options* meniju pod *Measurment* opcijom, i u *General* prozoru *Channel Usage*, podešen odgovarajući broj *CAN* kanala. U *Simulation Setup* (u ovom prozoru je ukupan sistem prikazan grafički sa mrežama, uređajima, i svim mrežnim čvorovima) prozoru su potom kreirana dva *ECU* čvora i jedan *I-Generator* (koristi se za generaciju *CAN* signala) koji su povezani na prethodno napravljenu *CAN* mrežu. U meniju *Hardware* pod opcijom *Network Hardware* je nameštena odgovarajuća brzina prenosa koja mora da odgovara brzini prenosa koja se koristi za dekodovanje. Ako ovo nije pravilno podešeno, dekodovanje neće biti ispravno.

Kao deo *CANoe, CANdb++ Editor* predstavlja okruženje za kreiranje baze podataka (formata *.dbc*) koja opisuje celokupnu *CAN* mrežu. Svaka *CAN* mreža je definisana sa: mrežom, upravljačkom jedinicom, promenljivama okruženja, mrežnim čvorovima, porukama i signalima.

Potom je kreirana baza podataka za *CAN* mrežu u *CANoe Candb++ Editor*-u unutar koje su kreirana 2 čvora (primarni i sekundarni). Nakon toga su kreirana dva signala, pri čemu je jedan rezervisan za slanje podataka od primarnog čvora do sekundarnog, a drugi za slanje obrnutim smerom. Zatim su ovi signali mapirani za odgovarajuću poruku koja je prethodno kreirana. Na kraju su ove poruke mapirane na svaki od odgovarajućih čvorova, čime se završava kreiranje baze podataka. Unutar *I-Generator*-a su kreirani signali koji će se slati od primarnog do sekundarnog čvora. Za pravljenje signala mora se odabrati kanal, identifikator, tip signala (da li je sa standardnim identifikatorom ili produženim), dužina podataka, a unutar *Raw Data* prozora se konfiguriše svaki od okteta za slanje.

Nakon podešavanja konfiguracije u *CANoe*-u i uključivanja napravljene baze podataka, urađeno je povezivanje elektronske upravljačke jedinice i računara *USB* kablom, a zatim je pokrenuta simulacija dok su sa druge strane povezane dve probe osciloskopa na *CAN-H* i *CAN-L* kanal čije su linije izvedene i priključene na prototipsku ploču (Sl. 2). Pre samog pokretanja konfiguracije, mora se potvrditi unutar *Home* prozora da je konfiguracija podešena u *Online Mode* (tada se podaci sa trenutnog merenja koriste – hardverski ili simulacijom, za razliku kada se koriste podaci iz jednog od fajlova za evidentiranje) i da se koristi *Real Bus* (za razliku od simulirane magistrale), pošto će se signal generisati upotrebom elektronske upravljačke jedinice. Praćenje signala u vremenu se vrši unutar *Data* prozora iz *Analysis* menija.

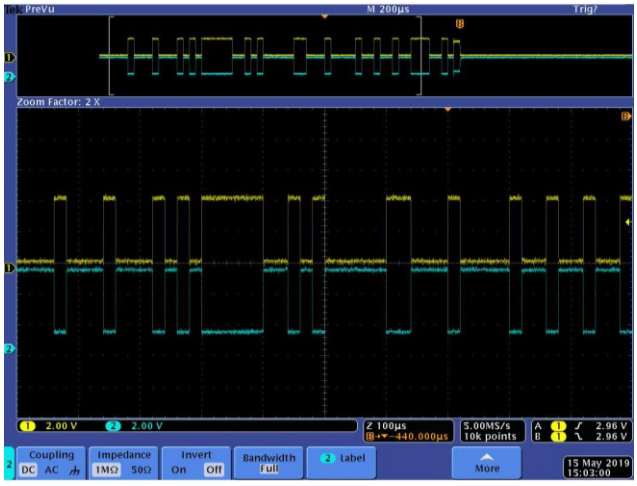


Sl. 2 Dijagram povezivanja *CAN*-a.

Nakon što je signal snimljen osciloskopom, uzorkuje se na vrednosti prethodno podešene horizontalne skale. Vrednosti koje imaju napon veći od postavljenog okidača se postavljaju na 1, u suprotnom se postavljaju na 0. Da bi se našao smisao niza jedinica i nula korišćen je *Knuth-Morris-Pratt* [4] algoritam za pretragu nizova koji traži pojavu obrasca unutar niza, ali kada dođe do neslaganja, obrazac se pomera za neki optimalan broj karaktera, tako zaobilazeći preispitivanje ranije usklađenih karaktera, sto ubrzava pretragu. Potom se izdvajaju svi relevantni okviri, vrši se konverzija dobijenih binarnih brojeva u heksadecimalnu vrednost i izračunava se ciklična provera redudanse. Ako se ova izračunata provera poklapa sa cikličnom proverom redudanse izdvojenom iz signala, glavnom programu se prosleđuje dekodovan *CAN* signal sa naznakom da je provera tačna, u suprotnom, prosleđuje se dekodovan *CAN* signal sa naznakom da provera nije tačna. Upotrebom *Knuth-Morris-Pratt* algoritma se detektuju lavinski biti i 5 istih dominantnih ili recesivnih bita. Tek nakon ovoga se moglo nastaviti sa dekodovanjem pri čemu se gleda 13. bit signala koji ukazuje na to da li je korišćen *CAN 2.0 A* ili *CAN 2.0 B,* što je važno odrediti da bi signal bio pravilno dekodovan.

Prilikom izrade ovog rada, za merenje napona signala korišćen je *Tektronix DPO 4104B* osciloskop. Ovaj osciloskop ima četiri nezavisna kanala.

Sl. 3 prikazuje generisani *CAN* signal snimljen na osciloskopu. U ovom radu su korišćeni jedan ili dva kanala osciloskopa u zavisnosti od protokola koji se koristi. Okidači su potom podešeni za nivoe izmerenog napona, horizontalnu i vertikalnu skalu. Na samom početku ovo podešavanje je bilo ručno, praćenjem uputstva za upotrebu ovog osciloskopa [5], potom automatski – komunikacijom preko *USB*-a upotrebom *API*-ja (engl. *Application programming interface*).

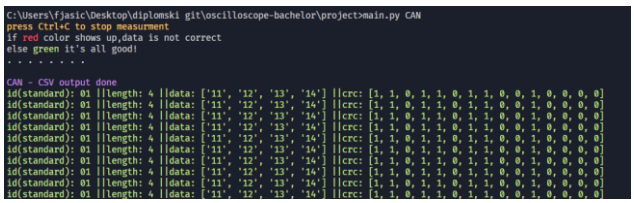


Sl. 3 Prikaz ekrana osciloskopa snimanja *CAN* signala.

Komunikacija sa osciloskopom je urađena upotrebom *PyVisa* Pajton modula. Kod je pisan praćenjem *PEP8* [6] konvencije. *PyVisa* omogućava inicijalizaciju osciloskopa kao objekata klase. Razmenjivanje podataka sa osciloskopom se svodi na dve metode klase. Praćenjem priručnika za programere od kompanije *Tektronix* [7] za korišćeni model osciloskopa poslate su komande kao parametar jednoj od dve prethodno opisane metode, u zavisnosti od namene (da li se želi podešavanje osciloskopa ili prikupljanje podataka). Poslate su komande za odabir kanala, postavljanje horizontalne i vertikalne skale, a potom za prikupljanje podataka nakon svake detekcije okidača.

IV. Rezultati

Uporednom metodologijom (poređenjem izlaza programskog rešenja i generisanog signala) se utvrdila ispravnost implementiranog dekodera. U slučaju *CAN* protokola, poređenjem dekodovane kontrolne sume snimljene osciloskopom i izračunate kontrolne dodatno se garantuje ispravnost implementiranog programskog rešenja. Sl. 4 predstavlja primer uspešnog izlaza dekodovanog signala na komandnoj liniji.



Sl. 4 Primer izlaza uspešnog dekodovanja *CAN*-a na komandnoj liniji.

testiranje je obavljeno sa 150 različitih signala (različiti okviri kako u varijaciji sadržaja okvira identifikatora i okvira podataka tako i u veličini okvira podataka) upotrebom elektronske upravljačke jedinice, pri čemu se posebno povelo računa da se šalju okviri za testiranje mehanizma za detekciju „lavinskih“ bitova u slučaju dekodovanja *CAN* signala. Uspešnim izvršavanjem ispitnih slučajeva potvrđena je funkcionalnost implementiranog dekodera.

V. Zaključak

Realizacijom dekodera olakšana je analiza najčešće korišćenih signala u automobilskoj industriji jer je podešavanje samog osciloskopa kao i analiza signala potpuno automatizovano. Samom automatizacijom smanjen je uticaj ljudske greške, koja može da ima značajan uticaj na rezultate naročito prilikom većeg broja repetativnog ponavljanja. Dalji razvoj će se fokusirati na realizaciju automatske detekcije korišćenog protokola.

Literatura

[1] *CAN Specification Version 2.0*, Robert Bosch GmbH, 1991, http://esd.cs.ucr.edu/webres/can20.pdf

[2] *ISO 11898-1, "Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signaling*, ISO, First edition, Geneva, Switzerland, 2003.

[3] *CANoe – The Multibus Development and Test Tool for ECUs and Networks*, 2019, https://assets.vector.com/cms/content/products/canoe/canoe/docs/Fact%20Sheets/CANoe_FactSheet_EN.pdf

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, "Introduction to Algorithms", Third Edition, The MIT Press, London, England, 2010.

[5] *MSO4000B and DPO4000B Series Digital Phosphor Oscilloscopes User Manual*, Tektronix, https://download.tek.com/manual/071281002web.pdf

[6] Guido van Rossum, Barry Warsaw, Nick Coghlan (2001, Jul), *PEP 8 Style Guide for Python Code*, https://www.python.org/dev/peps/pep-0008

[7] *MSO4000 and DPO4000 Series Digital Phosphor Oscilloscopes Programmer Manual*, Tektronix, https://download.tek.com/manual/077024801web.pdf

Abstract

One of the goals of this paper was to explain the implementation of recording, processing and decoding of protocol signals used in the automotive industry – CAN, using osciloscope. In addition, it was necessary to achieve the automation of osciloscope. The automated protocol decoding makes it easier to diagnose car networks by giving the user a detailed view of the entire system bus.

One solution for the automotive bus analysis tool using an osciloscope

Filip Jašić, Dejan Bokan, Zoran Marčeta