# Oporavak od stanja s pogreškom kod komunikacije najčešće korištenim protokolima u automobilskoj industriji

**Turjak, Josip**



**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU**

**ELEKTROTEHNIČKI FAKULTET**

**Sveučilišni studij**

# OPORAVAK OD STANJA S POGREŠKOM KOD KOMUNIKACIJE NAJČEŠĆE KORIŠTENIM PROTOKOLIMA U AUTOMOBILSKOJ INDUSTRIJI

**Diplomski rad**

**Josip Turjak**

**Osijek, 2017.**

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

# IZJAVA O ORIGINALNOSTI RADA

**Osijek, 27.09.2017.**

| | |
|---|---|
| **Ime i prezime studenta:** | Josip Turjak |
| **Studij**: | Diplomski sveučilišni studij Računarstvo |
| **Mat. br. studenta, godina upisa:** | D 810 R, 09.10.2015. |
| **Ephorus podudaranje [%]:** | 0 |

Ovom izjavom izjavljujem da je rad pod nazivom**: Oporavak od stanja s pogreškom kod komunikacije najčešće korištenim protokolima u automobilskoj industriji**

izrađen pod vodstvom mentora Doc.dr.sc. Ratko Grbić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

**Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada**

**Osijek, 22.09.2017.**

**Odboru za završne i diplomske ispite**

# Imenovanje Povjerenstva za obranu diplomskog rada

| | |
|---|---|
| **Ime i prezime studenta:** | Josip Turjak |
| **Studij, smjer:** | Diplomski sveučilišni studij Računarstvo |
| **Mat. br. studenta, godina upisa:** | D 810 R, 09.10.2015. |
| **OIB studenta:** | 72999077924 |
| **Mentor:** | Doc.dr.sc. Ratko Grbić |
| **Sumentor:** | |
| **Sumentor iz tvrtke:** | Kristina Tošeski |
| **Predsjednik Povjerenstva:** | Izv. prof. dr. sc. Marijan Herceg |
| **Član Povjerenstva:** | Doc.dr.sc. Mario Vranješ |
| **Naslov diplomskog rada:** | Oporavak od stanja s pogreškom kod komunikacije najčešće korištenim protokolima u automobilskoj industriji |
| **Znanstvena grana rada:** | **Telekomunikacije i informatika (zn. polje elektrotehnika)** |
| **Zadatak diplomskog rada:** | Najčešće korišteni komunikacijski protokoli u automobilskoj industriji su CAN, FlexRay i CANFD. Odlika ovih protokola je mogućnost prebacivanja upravljačke jedinice u stanje s pogreškom kada se na sabirnici pojavi greška u komunikaciji. Potrebno je istražiti i opisati mehanizme vraćanja iz stanja s pogreškom te opisano modelirati na Aurix Tricore procesoru uz injekciju pogreške prilikom komunikacije. Osim toga potrebno je testirati otpornost na greške u komunikacijskom sustavu za pojedini protokol i izvesti mjerenja ponašanja sustava pri različitim opterećenjima sabirnice. (sumentor Kristina Tošeski, Institut RT-RK Osijek, Osijek) |
| **Prijedlog ocjene pismenog dijela ispita (diplomskog rada):** | Izvrstan (5) |
| **Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:** | Primjena znanja stečenih na fakultetu: 3 bod/boda<br>Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda<br>Jasnoća pismenog izražavanja: 3 bod/boda<br>Razina samostalnosti: 3 razina |
| **Datum prijedloga ocjene mentora:** | 22.09.2017. |

| | |
|---|---|
| Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija: | Potpis: |
| | Datum: |

SADRŽAJ

# 1. UVOD

Današnji automobil sastoji se od nekoliko desetaka kontrolnih jedinica (engl. *Engine Control Unit*-ECU) koje omogućavaju sigurnu i ugodnu vožnju. Ove kontrolne jedinice zadužene su za poslove koji uključuju kontrolu dotoka goriva, pomoć prilikom parkiranja pa sve do mogućnosti elektronskog namještanja sjedala. Neki zadaci, poput aktiviranja zračnih jastuka, važniji su od drugih te kod njih nema mjesta za pogreške. Pogreška u komunikaciji između ECU-a u takvim situacijama može dovesti do materijalne štete, a u najgorem slučaju i do ljudskih žrtava. Zbog toga važno je unaprijed znati koje se sve pogreške mogu pojaviti te koje postupke treba poduzeti u slučaju da dođe do takvih situacija. Ovaj rad bavit će se upravo tom problematikom pri čemu je naglasak stavljen na dva najčešća komunikacijska protokola u automobilskoj industriji, a to su CAN i FlexRay protokoli.

Zadatak ovoga rada jest izazvati pogreške prilikom komunikacije CAN i FlexRay protokolima te vidjeti postoje li ugrađene funkcije za oporavak od istih. U slučaju da oporavak od pogreške za određenu situaciju ne postoji, treba se pronaći optimalno rješenje. Sam rad strukturiran je na sljedeći način. U drugom poglavlju opisane su osnove rada CAN i FlexRay protokola. Pogreške koje se pojavljuju u navedenim protokolima opisane su u trećem poglavlju dok se opis programskog rješenja nalazi u četvrtom poglavlju.

# 1 KOMUNIKACIJSKI PROTOKOLI U AUTOMOBILSKOJ INDUSTRIJI

U ovom poglavlju opisani su komunikacijski protokoli koje obuhvaća ovaj rad, a to su CAN i FlexRay protokol. Poglavlje sadrži sažet i jasan opis oba protokola u svrhu upoznavanja čitatelja s osnovama rada protokola.

## 2.1. CAN protokol

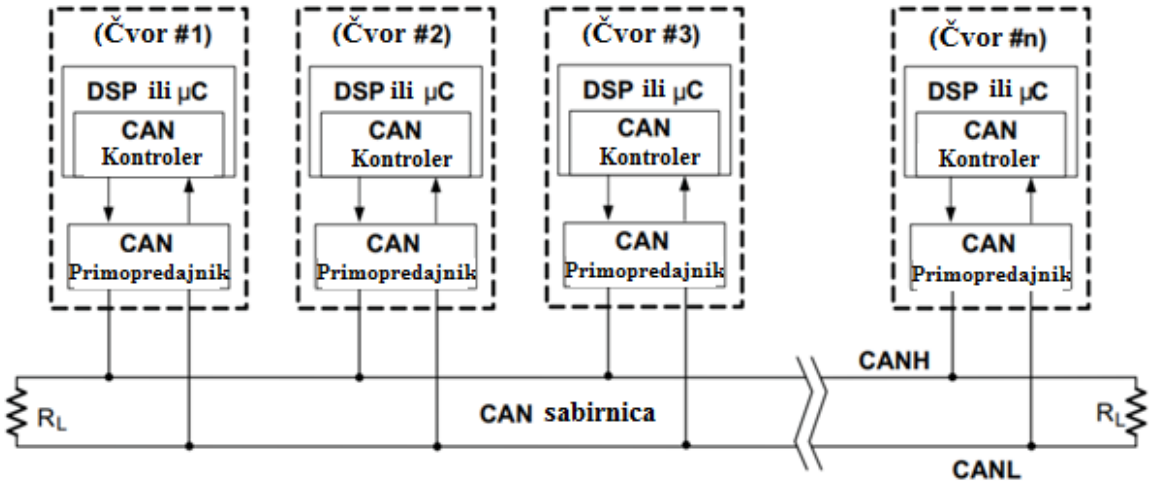
Razvoj ovog protokola započela je njemačka tvrtka Bosch početkom 1980.-ih te mu je dodijeljen naziv CAN (engl. *Controller Area Network*-CAN). CAN definira serijski prijenos podataka komunikacijskim kanalom, koji najčešće tvore par bakrenih žica, brzinama od 100 kilobita po sekundi (kb/s) do 1 megabit po sekundi (Mb/s). Standardiziran je pomoću četiri ISO dokumenta. ISO 11898-1 definira opće specifikacije CAN protokola te ga definira kao komunikaciju pogonjenu događajima (engl. *event-driven communication*). ISO 11898-2 dopušta brzine prijenosa podataka do 125 kb/s. Najčešće se koristi u dijelovima automobila koji nisu neophodni za vožnju (npr. elektronsko podešavanje sjedala i retrovizora ili pojačavanje i stišavanje zvuka na radiju). ISO 11898-3 definira prijenosnu brzinu do 1 Mb/s. On se koristi u dijelovima automobila koji su kritični za vožnju (npr. komunikacija između ECU-ova unutar motora i osovine automobila). Posljednji standard, ISO 11892-4, definira nadopunu sloja podatkovne veze (engl. *Data Link Layer*) koja dodaje opciju za komunikaciju pogonjenu vremenskim okidačima (engl. *Time-triggered communication*) [1].

### 1.1.1. Građa CAN mreže

CAN mreža sastoji se od čvorova (engl. *node*) koji su povezani s fizičkim medijem. Najčešći fizički medij je UTP sabirnica (engl. *Unshielded Twisted Pair*-UTP). Na krajevima CAN mreže nalaze se otpornici od 120 ohma za koji služe za onemogućavanje pojave refleksije poruka. Prema ISO 11898 standardu najveći broj čvorova u mreži je 32.

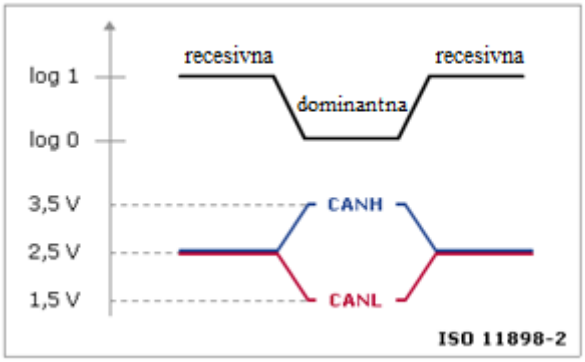
Svaki CAN čvor sastoji se od CAN primopredajnika (engl. *CAN transciever*) te CAN kontrolera (Sl. 2.1.). Uloga primopredajnika je povezivanje kontrolera s fizičkim prijenosnim medijem. Komunikacija između čvorova vrši se promjenom diferencijala napona na sabirnici koja se sastoji od dvije linije: CANH i CANL (engl. *CAN-high, CAN-low*). Na taj način rješava se problem smetnji koje se pojavljuju zbog rada automobila (ne uklanjaju se smetnje već se njime osigurava robusnost cjelokupnog sustava). Linije sabirnice su upletene kako bi se smanjio utjecaj induktivnih smetnji. Zbog ograničene brzine propagacije signala, povećava se mogućnost pojave refleksije signala. Kako bi se pojava refleksije u potpunosti izbjegla, koriste se otpornici za terminaciju (engl. *termination resistors*). Ključan dio kod takvih otpornika je tzv. impedancija

električne linije koja iznosi 120 ohm-a. Za prijenos podataka koristi se kodiranje bez vraćanja nule (engl. *Non Return Zero*-NRZ) koje pridružuje logičke simbole (logičku nulu i jedinicu) fizičkom signalu odnosno diferencijalu napona unutar CAN sabirnice. Problem kod ovakvog predstavljanja podataka jest mogućnost gubitka sinkronizacije u slučaju kada se šalje veliki broj bitova iste razine u nizu. U CAN protokolu taj problem je riješen uvođenjem metode umetanja dodatnog bita (engl. *bit stuffing*). Tom metodom riješen je problem sinkronizacije tako da se poslije svakih pet bitova jednake razine umetne jedan dodatni bit suprotne razine na čvoru koji šalje poruku, dok se na čvoru koji poruku prima, nakon svakih pet bitova jednake razine ispušta idući bit.



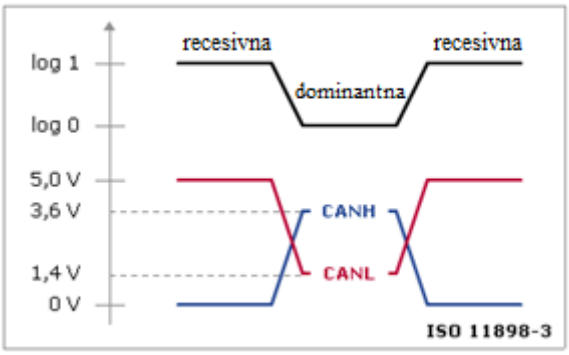

**Sl. 2.1.** Izgled CAN mreže [2].

Diferencijal napona na sabirnici ovisi o sučelju sabirnice koja (se koristi. Naime, postoji razlika između sučelja CAN sabirnice velike brzine (definiranim ISO 11898-2 standardom) i CAN sabirnice male brzine (definiranim ISO 11898-3 standardom). Kod sabirnica velike brzine logička jedinica se dodjeljuje diferencijalu napona koji iznosi 0 volta (V), a logička nula je dodijeljena diferencijalu napona od 2 V (Sl. 2.2.). Vrijednosti tih diferencijala napona vrijede u slučaju da nema vanjskih smetnji. Zbog mogućeg utjecaja smetnji u okolini sabirnice ISO 11898-2 standard tumači diferencijal napona iznad 0.9 volta kao dominantnu razinu odnosno logičku nulu.




**Sl. 2.2.** Naponske razine kod CAN sabirnice velike brzine [1].

Sabirnica malih brzina, prema ISO 11898-3 standardu, tumači diferencijal napona od 5 V kao logičku jedinicu te diferencijal napona od 2 V kao logičku nulu (Sl. 2.3.). Logička nula se uzima kao dominantna razina sabirnice, a recesivna razina sabirnice predstavlja logičku jedinicu. Pojava dominantne razine sabirnice prepisuje recesivnu razinu pa se CAN sabirnica zajedno s čvorovima može predstaviti kao logički sklop I.



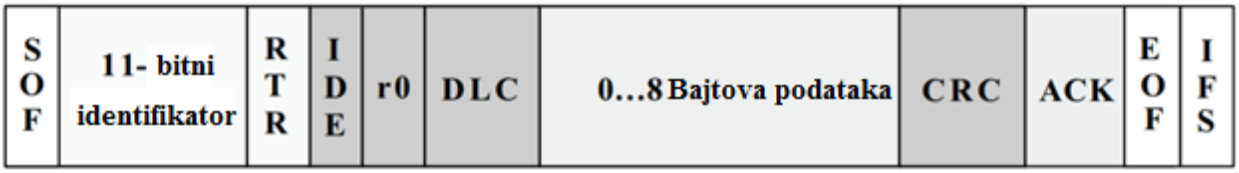

**Sl. 2.3.** Naponske razine kod CAN sabirnice male brzine[1].

Jedna od prednosti CAN mreže jest to što ne postoji centralni čvor koji bi imao dužnost dodjeljivanja prava pristupa sabirnici svakom čvoru posebno. Rješenje, koje je implementirano u CAN mrežu, je decentralizirani pristup sabirnici. Svaki čvor unutar mreže ima pravo kada se dogodi nekakav događaj, poslati poruku sabirnicom u bilo kojem vremenu. Sabirnica će biti zauzeta samo ako se šalju nove informacije. Svaka poruka može biti poslana na sve čvorove u mreži (engl. *broadcasting*), ali moguće je i odabrati koji čvorovi će primiti poruku. U slučaju da više čvorova pokušava istovremeno pristupiti sabirnici kako bi poslali poruku, dolazi do

arbitraže u kojoj se uspoređuju bitovi identifikacijskog polja poruke. Rezultat usporedbe jest dodjeljivanje prava pristupa sabirnici za slanje poruka čvoru čija je vrijednost navedenog polja najmanja. Najmanja vrijednost se traži iz razloga što je dominantna razina sabirnice logička nula i samim time sabirnici će dobiti pristup CAN čvor s najviše logičkih nula zaredom u identifikatoru polja počevši od najznačajnijeg bita (engl. *most significant bits*). Čvorovi koji su izgubili u arbitraži prebacuju se na stanje primanja poruka i čekaju ponovno oslobađanje sabirnice.

### 1.1.2. Izgled CAN poruke

Struktura poruke koja se odašilje preko CAN sabirnice CAN mrežom je jasno definirana, ali ima i par izuzetaka. Polja bitova (engl. *bit fields*) od kojih se sastoji standardna CAN poruka vidljivi su na slici 2.4.




**Sl. 2.4.** Izgled standardne CAN poruke [2].

Prvo polje označava početak poruke (engl. *Start Of Frame*-SOF) i njegova svrha je sinkronizacija čvorova na sabirnici koja je bila u stanju mirovanja (engl. *idle*). Drugo polje je identifikator kojega čini 11 bitova čija vrijednost predstavlja prioritet poruke. Što je njegova vrijednost manja to je prioritet poruke veći. Poslije identifikatora slijedi polje zahtjev za udaljeni prijenos (engl. *Remote Transmission Request*-RTR). RTR polje je postavljeno na dominantnu vrijednost kada je potrebna informacija od drugog čvora. Svi čvorovi primaju zahtjev za zatraženom informacijom, ali se po identifikatoru zahtjeva zna koji čvor treba odgovoriti. Odgovor se također šalje svim čvorovima i njegove informacije mogu koristiti svi čvorovi koji su zainteresirani za njih. Iza RTR polja nalazi se identifikacijsko produženje (engl. *Identifier Extension*-IDE). U slučaju da je ono postavljeno na dominantnu razinu (logičku nulu), znači da ne postoji produženje identifikacijskog polja odnosno radi se o standardnom formatu CAN poruke. U slučaju da IDE polje ima recesivnu vrijednost odnosno logičku jedinicu, to implicira da se u nastavku poruke nalazi dodatno identifikacijsko polje od 18 bitova koje se koristi u produženom formatu CAN poruke (engl. *extended format*). Sljedeći dio poruke je rezervirani bit za potrebe dodatnih funkcionalnosti u narednim verzijama protokola. Polje iza rezerviranog bita

se sastoji od četiri bita i predstavlja duljinu podataka koji će se poslati (engl. *Data Length Code*). Slijedi podatkovno polje od osam bajtova koje je zaštićeno sa šesnaest bitnom zaštitom cikličke redundancije (engl. *Cyclic Redundancy Check*-CRC). Iza CRC zaštite dolazi polje od dva bita čija je svrha potvrđivanje ispravnosti poruke (engl. *Acknowledge*-ACK). Ovo polje se sastoji od jednog bita koji je postavljen na recesivnu vrijednost. U slučaju da je primljena poruka bez pogrešaka, recesivna vrijednost prvog bita ACK polja se prepisuje dominantnom vrijednošću. Ako je došlo do pogreške prilikom slanja poruke, prvi bit ACK polja ostaje recesivan te se poruka odbacuje i čvor koji šalje poruku započinje ponovno slanje. Drugi bit polja je granični bit, odnosno dio koji odvaja ACK polje od polja koje slijedi. Nakon ACK polja dolazi do završetka slanja koje je određeno sedam bitnim poljem zvanim kraj okvira (engl. *End Of Frame*-EOF). Iako nije dio CAN poruke, važno polje kod razmjene poruka CAN mrežom je međuokvirni prostor (engl. *Interframe Space*-IFS). Njegova vrijednost predstavlja vrijeme koje je potrebno CAN kontroleru da premjesti primljeni okvir (primljen bez greške) na njegovu poziciju u podatkovnom međuspremniku.

Postoje četiri vrste poruka koje se mogu odašiljati unutar CAN mreže, a razlika u njihovom izgledu je okvir koji se koristi za njihovo slanje. Vrste poruka (okvira) su: podatkovni okvir (engl. *data frame*), udaljeni okvir (engl. *remote frame*), okvir za pogrešku (engl. *error frame*) te okvir preopterećenja (engl. *overload frame*).

Podatkovni okvir je najčešći tip poruke. On se sastoji od arbitracijskog polja (ID polje i RTR polje), podatkovnog polja, CRC zaštite i ACK polja. Podatkovni okvir se koristi za prijenos korisničkih podataka.

Udaljeni okvir koristi se da se zatraže podaci od određenog CAN čvora unutar mreže. Ovaj okvir ne koristi se često u autoindustriji, jer se komunikacija ne temelji na zahtjevima već na odašiljanju informacija samoinicijativno. Za razliku od podatkovnog okvira, udaljeni okvir nema podatkovno polje te je RTR polje postavljeno na recesivnu vrijednost.

Okvir za pogrešku je posebna vrsta poruke koja se ne pridržava standardnih pravila formatiranja CAN poruka. Kada se pojavi pogreška u poruci, šalje se okvir za pogrešku od strane svih čvorova dok čvor koji je poslao poruku s pogreškom započinje ponovno slanje poruke. U slučaju da se pogreška nastavi pojavljivati, postoje brojila pogrešaka koja ne dopuštaju neometano i kontinuirano slanje takvih poruka. Više o pojavljivanju i rukovanju pogreškama bit će objašnjeno u zasebnom poglavlju ovoga rada [1].

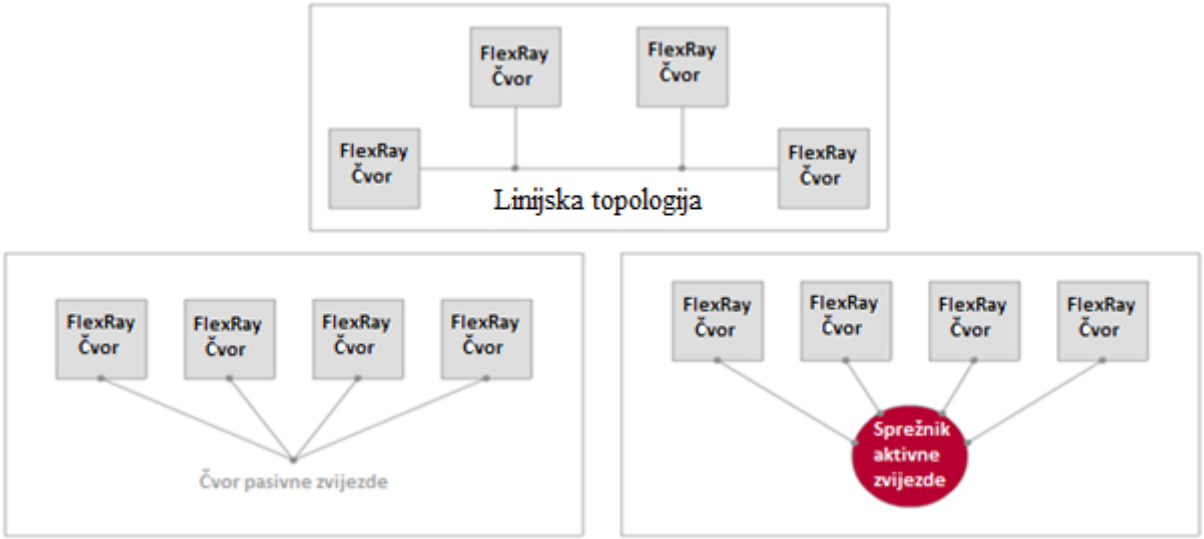
## 1.2. FlexRay protokol

Razvoj automobilske industrije uključuje poboljšavanje sigurnosti, smanjivanje štetnosti za okoliš, povećanje komfora i sigurnosti. Da bi taj razvoj bio održiv, brzina, pouzdanost i količina prijenosa podataka između ECU-a unutar automobila mora se povećati. U tu svrhu započeo je razvoj FlexRay protokola. FlexRay protokol koristi se za komunikaciju između ECU-a koji su od kritične važnosti za omogućavanje sigurne i udobne vožnje. Takvi ECU-i zaduženi su za elektronsku kontrolu stabilnosti ( engl. *Electronic Stability Control*-ESC) i za pravovremeno napuhavanje zračnih jastuka. FlexRay je protokol pogonjen vremenskim okidačima (engl. *Time triggered event*) što znači da se prijenos podataka vrši prema predefiniranom rasporedu. Takva komunikacija ne dopušta čvorovima da pristupaju komunikacijskoj sabirnici u bilo kojem trenutku, kao primjerice kod CAN protokola. Metoda koja omogućuje ovakav protokol jest metoda raspodjele vremena uz višebrojni pristup (engl. *Time Division Multiple Access*-TDMA). TDMA se bazira na komunikacijskom rasporedu sastavljenom od određenog broja statičkih vremenskih polja jednake duljine (engl. *static slots*) koja se pridružuju FlexRay čvorovima. Čvor kojemu je pridružen trenutno vremensko polje ima pravo slati poruke preko sabirnice. Sva vremenska polja se odašilju periodički odnosno deterministički Komunikacijski raspored je zapravo komunikacijski ciklus. No, kako periodično slanje podataka nije dobro rješenje u svim situacijama, FlexRay protokol definira i dodatnu opciju dinamičkog segmenta poruka. Pomoću te opcije poruke se mogu slati i bez pridržavanja vremenskog ciklusa odnosno komunikacija postaje pogonjena događajima (engl. *Event-triggered*) . Brzina prijenosa je do 10 Mb/s što je 10 puta više od CAN protokola. Uz to, FlexRay je dizajniran da podržava jednokanalnu i dvokanalnu konfiguraciju. Dvokanalna konfiguracija omogućava prijenosne brzine i do 20 Mb/s (dva kanala po 10 Mb/s) ili pak jedan kanal služi kao pričuva odnosno redundancija. Implementacija dvokanalne konfiguracije danas je skuplja varijanta i često nepotrebna, ali daljnjim razvojem tehnologije vjerojatno će imati više primjena. FlexRay sabirnicu čini nezaštićeni, upleteni par bakrenih žica kao i kod CAN protokola, a na krajevima se nalaze otpornici koji služe za terminaciju odnosno onemogućavanje refleksije signala [3].

### 1.2.2. Građa FlexRay mreže

Postoji više načina umrežavanja čvorova unutar FlexRay mreže, a najčešći su: linijsko umrežavanje te umrežavanje aktivnom ili pasivnom zvijezdom. Kod linijskog umrežavanja čvorovi su spojeni na jednu ili dvije komunikacijske sabirnice (druga je redundantna) koje ne smiju prelaziti duljinu od 24 metra između svakog čvora (ako se želi postići maksimalna brzina prijenosa od 10 Mb/s). Prema FlexRay specifikacijama, maksimalni broj čvorova na jednoj liniji

iznosi 22. U slučaju da se mreža temelji na pasivnoj zvijezdi, čvorovi su međusobno povezani pomoću tzv. čvora pasivne zvijezde. Pomoću ovog načina umrežavanja broj čvorova i razmak između njih ostaju isti kao i kod linijskog umrežavanja (24 metra između čvorova kojih maksimalno može biti 22). Alternativa umrežavanju pasivnom zvijezdom je umrežavanje aktivnom zvijezdom u kojoj su čvorovi međusobno povezani pomoću sprežnika aktivne zvijezde (engl. *active star coupler*). Sprežnik služi za prihvaćanje signala od ostalih čvorova, njihovo pojačavanje i distribuciju svim ostalim komunikacijskim sabirnicama. Postoji samo jedan čvor na svakom kraju komunikacijske sabirnice. Kao i kod prijašnjih načina umrežavanja, maksimalna udaljenost između dva spojena čvora je 24 metra. Prednost ovakvog načina umrežavanja jest to što sprežnik odspoji neispravne komunikacijske sabirnice te se tako izbjegava širenje pogrešaka. Nedostatak ove opcije je taj što postoji kašnjenje prilikom prenošenja signala, jer sprežnik treba određeno vrijeme potrebno za postizanje operativnog stanja (najviše 450 nanosekundi). Sva tri navedena načina umrežavanja vidljiva su na slici 2.5.

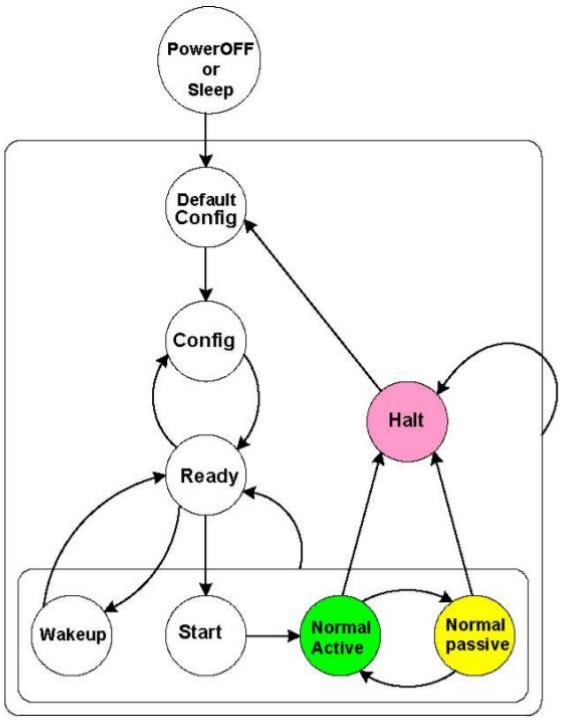



**Sl. 2.5.** Načini umrežavanja kod FlexRay mreže. Linijsko umrežavanje gornja slika, dolje lijevo umrežavanje pasivnom zvijezdom te dolje desno umrežavanje sprežnikom aktivne zvijezde [3].

FlexRay čvor sastoji se od komunikacijskog kontrolera tzv. FlexRay upravljača te od pogonskih programa sabirnice (engl. *bus drivers*) tzv. FlexRay primopredajnika. FlexRay upravljač zadužen je za izradu okvira poruka, omogućavanje pristupa sabirnici, uočavanje i rukovanje greškama, sinkronizaciju, buđenje i uspavljivanje FlexRay sabirnice te šifriranje odlaznih i dešifriranje dolaznih poruka. FlexRay primopredajnik služi za spajanje upravljača sa sabirnicom i njegova glavna zadaća je pretvorba signala. Pretvorba se vrši iz logičkog signalnog niza, primljenog od

strane upravljača, u fizički signal koji se šalje na sabirnicu. Pretvorba uključuje i obrnuti proces kada se radi o primanju poruke.

FlexRay upravljač ima 8 različitih stanja u kojima se može nalaziti ovisno o napretku u komunikaciji. Sva stanja i mogući prijelazi iz jednog stanja u drugo vidljivi su na slici 2.6.




**Sl. 2.6.** Prikaz stanja FlexRay upravljača i mogućnosti prelaska iz jedno u drugo stanje [4].

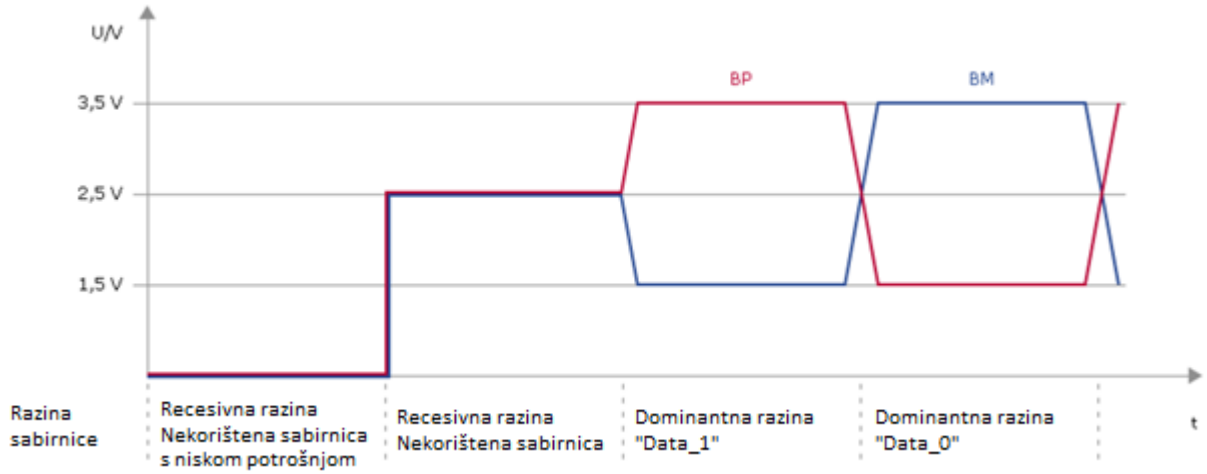
Detaljniji opis stanja je sljedeći:

1. „Default config“- postavljanje predefiniranih vrijednosti,
2. „Config“- inicijalizacija početnih postavki koje uključuju komunikacijski ciklus i brzinu prijenosa,
3. „Ready“- definiranje unutarnjih komunikacijskih postavki,
4. „Wakeup“- buđenje čvora koji trenutno ne komunicira,
5. „Start“- započinje sinkronizacija sata i priprema se za komunikaciju,
6. „Normal active/ passive“- neometana komunikacija,
7. „Halt“- komunikacija je prestala.

Prijenos signala preko komunikacijske sabirnice temelji se na promjeni diferencijala napona slično kao kod CAN protokola. Sabirnica se sastoji od dvije linije: sabirnica plus (engl. *bus plus*-

BP) i sabirnica minus (engl. *bus minus*-BM) koje se isprepliću kako bi se smanjio utjecaj smetnji. Na osnovu specifikacija fizičkog električnog sloja (engl. *Electrical Physical Layer Specification*) definiraju se četiri razine sabirnice koje se dodjeljuju recesivnom ili dominantnom stanju sabirnice. Recesivno stanje je karakterizirano diferencijalom napona od 0 V dok se dominantno stanje pripisuje diferencijalu napona većem od 0 V. Prve dvije recesivne razine su: nekorištena sabirnica (engl. *Idle bus level*) i nekorištena sabirnica s niskom potrošnjom energije (engl. *Idle low power bus level*). Razina nekorištene sabirnice se postiže kada je električni potencijal obje linije (BP i BM) jednak 2.5 V što rezultira diferencijalom napona u vrijednosti od 0 V. Valjan raspon električnog potencijala kod ove razine je između 1.8 V i 3.2 V. Razina nekorištene sabirnica s niskom potrošnjom energije se pojavljuje kada su svi FlexRay primopredajnici u stanju niske potrošnje energije. Diferencijal napona iznosi 0 V, a valjan raspon električnog potencijala kod ove razine je između -0.2 V i 0.2 V.

Dominantne razine sabirnice označavaju se s Data_0 i Data_1. Kod Data_0 razine BP linija ima električni potencijal od 1.5 Volta dok BM linija ima potencijal od 3.5 Volta. Diferencijalni napon ove razine iznosi -2 Volta. Data_0 razina sabirnice predstavlja logičku nulu. Data_1 razinu karakterizira diferencijal napona u iznosu od 2 Volta, koji je rezultat razlike električnog potencijala linije BP od 3.5 Volta i električnog potencijala linije BM od 1.5 Volta. Ovom razinom predstavlja se logička jedinica. Razine FlexRay sabirnice nalaze se na slici 2.7 [3].




**Sl. 2.7.** Razine FlexRay sabirnice [3].

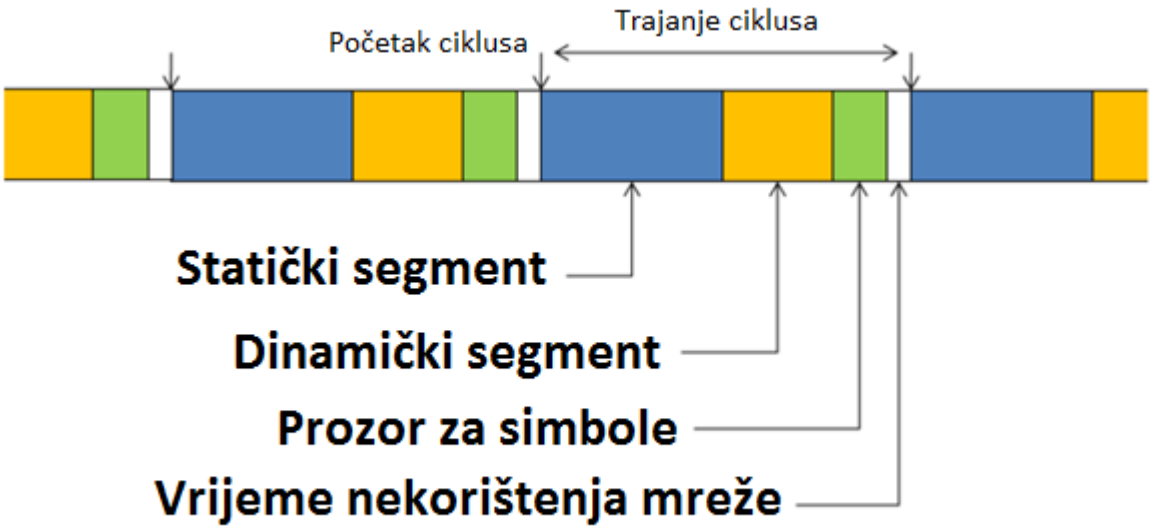
### 1.2.3. Komunikacijski ciklus

Kao što je ranije u radu navedeno, FlexRay komunikacija se temelji na vremenski pogonjenim okidačima, ali postoji dodatna metoda kojom se omogućuje komunikacija pogonjena događajima. TDMA metoda se temelji na komunikacijskom rasporedu prema kojemu se statička vremenska polja dodjeljuju čvorovima. Kako bi se omogućila komunikacija pogonjena događajima upotrebljava se metoda fleksibilne podjele vremena uz višebrojni pristup (engl. *Flexible Time Division Multiple Access*-FTDMA) čija je osnova TDMA metoda. FTDMA metodu karakterizira kombinacija statičkih i dinamičkih segmenata unutar komunikacijskog ciklusa.

Statički segment se sastoji od statičkih polja jednake duljine koji služe za slanje statičkih poruka tijekom periodične komunikacije. Uvjet za slanje takvih poruka je sinkronizacija lokalnih brojača koji predstavljaju specifičnu statičku poruku i određeni FlexRay čvor. Unutar statičkog segmenta moguće je definirati 1023 statičkih polja. Svako statičko polje ima četiri dijela. Prvi dio je odstupanje akcijske točke (engl. *action point offset*). Akcijska točka je trenutak u kojemu se započinje prijenos poruke, a njezino odstupanje služi kako bi se izvršila sinkronizacija signala takta FlexRay čvorova koji sudjeluju u komunikaciji. Nakon odstupanja akcijske točke slijedi FlexRay poruka na čijem kraju se nalazi tzv. graničnik kanala u mirovanju (engl. *channel idle delimiter*) koji služi kao pokazatelj kraja FlexRay poruke. Na kraju statičkog segmenta nalazi se dio koji predstavlja pauzu čija je duljina određena s odstupanjem akcijske točke, a naziva se kanal u mirovanju (engl. *channel idle*).

Dinamički segment je neobavezan dio komunikacijskog ciklusa koji služi za slanje događajima pogonjenih poruka. Ovaj segment dolazi uvijek iza statičkog segmenta. Dinamički segment započinje uvećavanjem lokalnih brojača FlexRay čvorova. Vrijednosti tih brojača predstavljaju specifičnu dinamičku poruku i određeni FlexRay čvor. U slučaju da ne postoji zahtjev od strane FlexRay čvora za slanje dinamičke poruke koja se podudara s trenutnom vrijednosti brojača, brojači FlexRay čvorova se uvećavaju za točno jednu duljinu malog polja (engl. *mini slot)* od kojega se sastoji dinamički segment. U slučaju da zahtjev postoji, FlexRay čvor zadužen za slanje dinamičke poruke odašilje poruku. Nakon odašiljanja čeka se da prođe jedno malo polje koje signalizira FlexRay čvorovima da uvećaju svoje brojače. Taj proces se ponavlja sve do kraja dinamičkog segmenta. Dinamičko polje slično je statičkom polju, jedina razlika je to što je moguće slati podatke različitih duljina i to što sadrži posljednji dinamički niz (engl. *dynamic trailing sequence*) kojim se osigurava prekid slanja dinamičke poruke pojavljivanjem nove akcijske točke u ciklusu.

Komunikacijski ciklus sastoji se od najmanje dva, a najviše četiri segmenta (vidi Sl. 2.8.). Segmenti koji su uvijek prisutni su statički segment i vrijeme nekorištenja mreže (engl. *Network Idle Time*-NIT). Dodatak u ciklusu može biti dinamički segment i prozor za simbole (engl. *symbol window*). Prozor za simbole služi za slanje simbola koji uključuju: simbol za buđenje FlexRay čvora, simbol za izbjegavanje preklapanja poruka i druge.




**Sl. 2.8.** Komunikacijski ciklus sa svim segmentima [5].

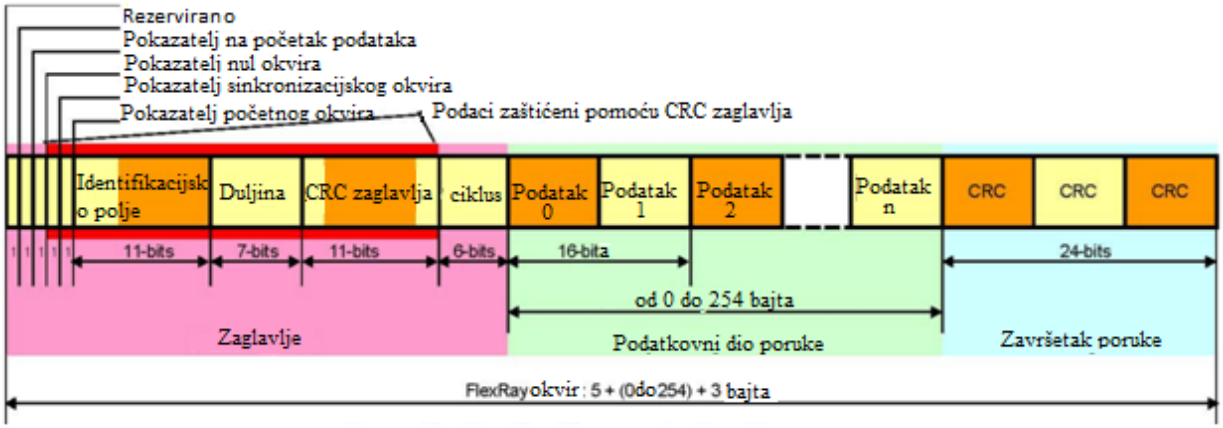
Trajanje jednog komunikacijskog ciklusa je definirano makrotikovima (engl. *macroticks*) koji su dodijeljeni pojedinom segmentu (duljina makrotika može iznositi od 1 ms do 5 ms iako je najčešća vrijednost 1 ms). Svaki makrotik se sastoji od određenog broja mikrotikova (engl. *microticks*) koji predstavljaju najmanju jedinicu vremena unutar lokalnih satova FlexRay čvorova.

Sinkronizacija lokalnih signala takta se temelji na tome da su vremena slanja i primanja svih statičkih poruka poznata svim FlexRay čvorovima. To osigurava mogućnost svakog FlexRay čvora da podesi svoje vremensko odstupanje i brzinu prijenosa. U FlexRay mreži barem dva FlexRay čvora služe kao sinkronizacijski čvorovi koji odašilju sinkronizacijsku poruku unutar statičkog polja u svakom komunikacijskom ciklusu. Svi FlexRay čvorovi uspoređuju svoje vrijeme i vrijeme koje se nalazi unutar sinkronizacijske poruke. Na osnovu te usporedbe

računaju korekcijsku vrijednost odstupanja vremena. Istom procedurom prilagođavaju se brzine prijenosa.

### 1.2.4. Izgled FlexRay poruke

Svaka FlexRay poruka se sastoji od tri dijela: zaglavlja (engl. *header*), podatkovnog dijela (engl. *payload*) te završnog dijela (engl. *trailer*). Zaglavlje zauzima 40 bita, podaci za slanje zauzimaju do 254 bajta i završni dio zauzima 24 bita (Sl. 2.9.).




**Sl. 2.9.** Izgled cjelokupne FlexRay poruke [4].

Na početku zaglavlja nalazi se pet bitova:

1. rezervirani (engl. *Reserved*) - bit rezerviran za buduću upotrebu,
2. pokazatelj na početak podataka (engl. *Payload preamble indicator*) - služi za pokazivanje mjesta na kojemu se nalaze vektori informacija o upravljanju mrežom,
3. pokazatelj nul okvira (engl. *Null frame indicator*) - pokazatelj ispravnosti poruke,
4. pokazatelj sinkronizacijskog okvira (engl. *Sync frame indicator*) - pokazuje da li se sinkronizacijski okviri koriste za sinkronizaciju unutar statičkog segmenta komunikacijskog ciklusa, i
5. pokazatelj početnog okvira (engl. *Start-up frame indicator*) - pokazuje je li FlexRay čvor koji šalje poruku ujedno i čvor koji započinje komunikaciju.

Ostatak zaglavlja čini 11 bitno identifikacijsko polje koje služi za prepoznavanje poruke i pridodjeljivanje iste određenom polju komunikacijskog ciklusa. Poslije identifikacijskog polja slijedi 7 bitno polje koje predstavlja duljinu podatkovnog dijela poruke. Zatim dolazi polje duljine 11 bita koje je zaduženo za CRC zaštitu zaglavlja. Na kraju FlexRay poruke se nalazi

brojač ciklusa koji predstavlja broj komunikacijskog ciklusa u kojemu je poruka poslana. Brojač može imati vrijednost od 0 do 63, a zauzima 6 bita.

Dio poruke u kojemu se nalaze podaci za prijenos zauzima do 254 bajta. Na kraju poruke nalazi se CRC zaštita koja služi kako bi se otkrile moguće nepravilnosti unutar podatkovnog dijela poruke.

# 2. POGREŠKE U KOMUNIKACIJI CAN I FLEXRAY PROTOKOLIMA

U ovom poglavlju opisane su sve pogreške koje se mogu pojaviti prilikom komunikacije s CAN i FlexRay protokolima.

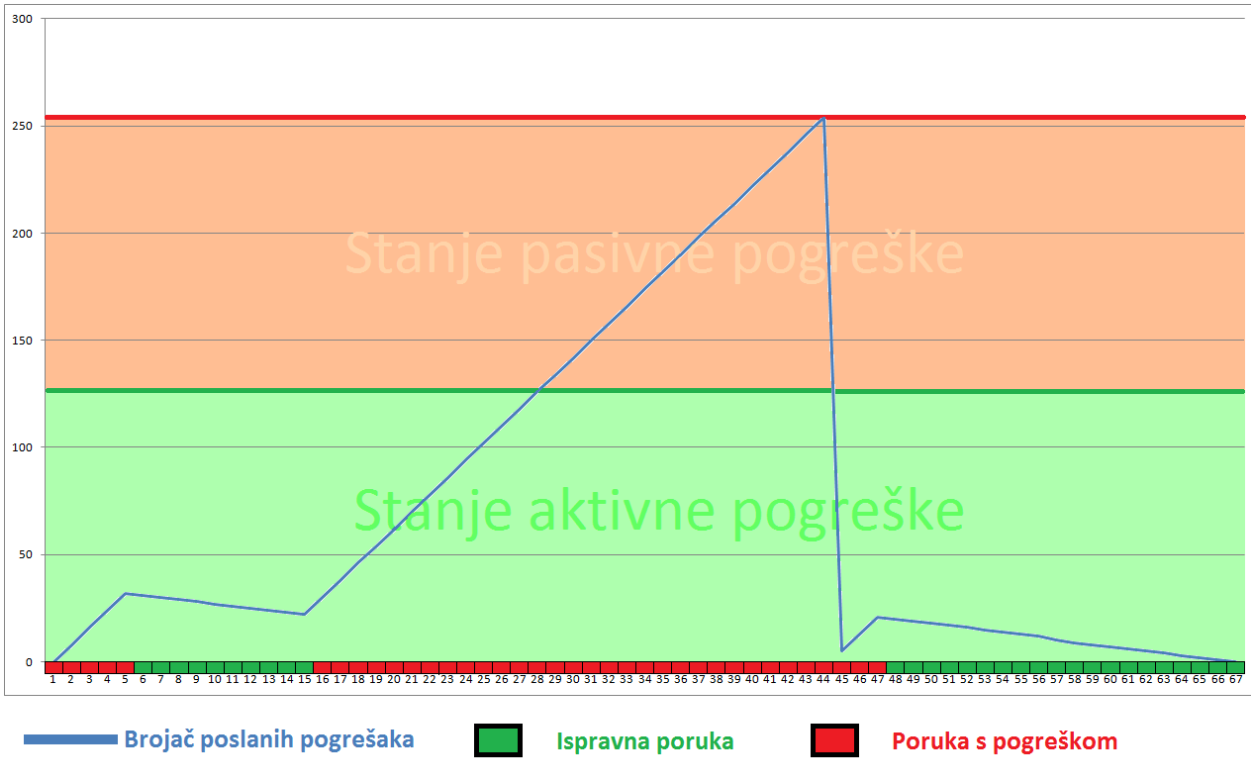
## 2.1. Pogreške u CAN protokolu

Robusnost CAN protokola vidljiva je u njegovoj mogućnosti rukovanja pogreškama u komunikaciji. Cilj rukovanja pogreškama je uočiti pogrešku u porukama na CAN sabirnici te omogući ponovno slanje takvih poruka od strane čvora koji takve poruke izvorno šalje. Ako se pronađe pogreška unutar poruke, čvor koji ju je pronašao šalje zastavicu pogreške (engl. *Error flag*) s kojom zauzima sabirnicu odnosno blokira ostatak prometa. Zastavica pogreške nalazi se unutar okvira za pogrešku. Postoje dvije vrste zastavica za pogrešku, a one su direktno vezane za trenutno stanje u kojemu se CAN čvor nalazi. Postoje tri različita stanja CAN čvorova, a to su:

1. Stanje aktivne pogreške (engl. *Error-active state*),
2. Stanje pasivne pogreške (engl. *Error-passive state*), i
3. Stanje isključene sabirnice (engl. *Bus-off*).

U slučaju da se čvor nalazi u stanju aktivne pogreške, zastavica pogreške sastojat će se od 6 dominantnih bitova u nizu što direktno krši pravilo umetanja dodatnog bita. Svi čvorovi prepoznaju pogrešku i odbacuju takvu poruku. Čvor koji je poslao poruku s pogreškom pokušat će ju poslati opet i tako se proces slanja ponavlja. Kako bi se ograničilo neprestano slanje poruke s pogreškom svaki čvor unutar CAN mreže ima dva brojača pri čemu jedan broji poslane poruke s pogreškom (engl. *Transmit Error Counter*-TEC) dok drugi broji primljene poruke s pogreškom (engl. *Receive Error Counter*-REC). Kada brojila prijeđu određene vrijednosti, stanje čvora se mijenja. U slučaju da je bilo koji od brojača prešao vrijednost od 128, čvor prelazi u stanje pasivne pogreške. U tom stanju CAN čvor još uvijek normalno može slati poruke, ali u slučaju pogreške šalje zastavice pogreške koje se sastoje od 6 recesivnih bitova u nizu. Na taj način izbjegava se blokiranje CAN sabirnice koje se događa kada se pošalje zastavica pogreške iz stanja aktivne pogreške. Ako se pogreške nastave pojavljivati, vrijednost brojača će se uvećavati sve do vrijednosti 255 kada se CAN čvor isključuje iz mreže. U takvom stanju on nema mogućnost slanja ni primanja poruke.

Brojač poslanih poruka s pogreškom se uvećava za 8 u slučaju da CAN čvor pošalje neispravnu poruku, dok se na čvorovima koji tu poruku primaju, brojač primljenih poruka s pogreškom uvećava za 1. U slučaju da se pošalje ispravna poruka, brojači se umanjuju za 1 (sl. 3.1.).




**Sl. 3.1.** Prikaz ovisnosti pojavljivanja pogrešaka i stanja brojača poslanih pogrešaka.

Postoji pet vrsta pogrešaka koje se mogu pojaviti u porukama na CAN sabirnici. Dvije pogreške su na razini bitova poruke, a tri pogreške su na razini cjelokupne poruke:

1. pogreška bitova (engl. *Bit Error*),
2. pogreška umetanja dodatnog bita (engl. *Stuff Error*),
3. pogreška potvrde (engl. *Acknowledge Error*),
4. pogreška forme (engl. *Form Error*), i
5. pogreška u CRC zaštiti (engl. *CRC Error*).

Pogreška bitova se pojavljuje u dva slučaja. Prvi slučaj je da CAN čvor pošalje dominantni bit, a na CAN sabirnici se pojavi recesivni bit. U drugom slučaju CAN čvor pošalje recesivni bit, a na CAN sabirnici se pojavi dominantni bit. U drugom slučaju neće doći do pojavljivanje pogreške tijekom provjere polja za arbitražu (ID, RTR i IDE) te polja za potvrdu primitka poruke (ACK). Pomoću ove pogreške osigurava se otkrivanje svih pogrešaka na strani pošiljatelja.

Pogreška umetanja dodatnog bita nastaje ako se prekrši pravilo o umetanju dodatnog bita. Između početka poruke (SOF) i kraja CRC dijela poruke ne smije biti više od pet bitova jednake

razine. Ako takav niz postoji onda je došlo do pogreške umetanja dodatnog bita što rezultira stvaranjem okvira za pogrešku te se poruka šalje ponovno.

Pogreška potvrde nastaje u slučaju kada niti jedan čvor nije primio poslanu poruku. U CAN protokolu svaku primljenu poruku primatelji moraju potvrditi postavljanjem ACK polja na dominantnu vrijednost (polje ima recesivnu vrijednost tijekom slanja). Ako ta vrijednost ostane recesivna niti jedan čvor unutar CAN mreže nije primio poruku i na taj način stvara se pogreška potvrde.

Pogreška forme je rezultat pojavljivanja dominantnog bita unutar određenih segmenata poruke koji uvijek imaju iste recesivne vrijednosti. Ti segmenti su: granični bitovi CRC i ACK polja te EOF polje.

Pogreška kod CRC zaštite nastaje ako je došlo do promjene unutar početka poruke sve do kraja podatkovnog polja. CRC se računa za taj dio poruke te se šalje unutar CRC polja. Na CAN čvoru koji prima poruku dolazi do ponovnog računanja očekivanog CRC koda te usporedbe s primljenim kodom. Ako se ta dva koda ne podudaraju, došlo je do pogreške CRC zaštite.

## 2.2. Pogreške u Flexray protokolu

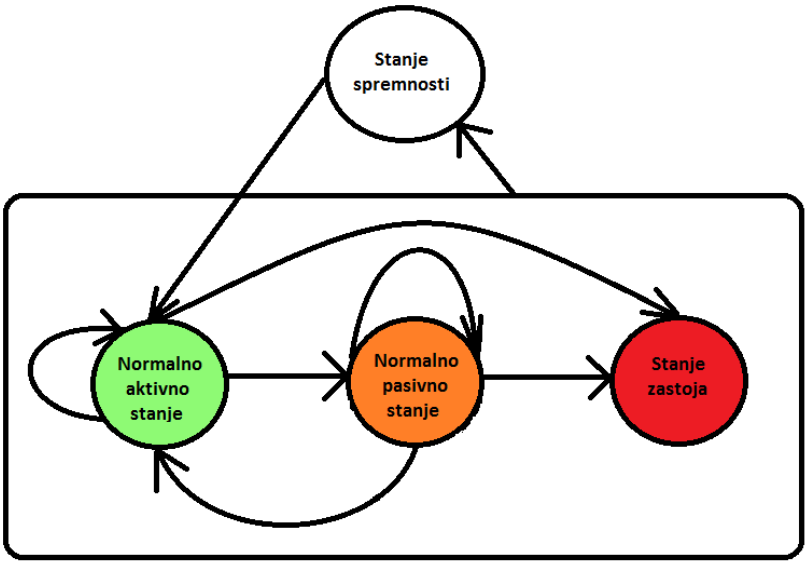
FlexRay protokol koristi se za komunikaciju između ECU-a od kritične važnosti unutar automobila. Ako do pogreške u komunikaciji ovim protokolom dođe onda se ona mora otkriti u što kraćem vremenu te se treba izvršiti oporavak od iste na najefikasniji način kako ne bi došlo do nesreće. Slično kao i kod CAN protokola, u FlexRay protokolu postoje 3 stanja pogreške čvorova prema [6]:

1. Normalno aktivno stanje (engl. *Normal active state*),
2. Normalno pasivno stanje (engl. *Normal passive state*), i
3. Stanje zastoja (engl. *Halt state*).

Da bi FlexRay čvor došao u normalno aktivno stanje, on mora prvo biti u stanju spremnosti (engl. *Ready*) u kojem se nalazi ako je uspješno napravljena početna konfiguracija čvora. Čvor koji se nalazi u normalnom aktivnom stanju može slati i primati poruke od ostalih čvorova unutar FlexRay mreže neometano. U tom stanju sinkronizacija s ostalim čvorovima je u potpunosti valjana i nastavlja se održavati.

U normalnom pasivnom stanju čvor gubi mogućnost slanja poruka, ali još uvijek ima mogućnost primanja poruka od ostalih čvorova FlexRay mreže. U ovom stanju čvor može biti nesinkroniziran s ostalim čvorovima i u tom slučaju on se nastoji sinkronizirati. Ako se uspije sinkronizirati vratit će se u normalno aktivno stanje.

U stanje zastoja FlexRay čvor ulazi iz bilo kojeg stanja čvora ako se stanje ručno promjeni (direktna promjena pomoću softverskog rješenja) ili se određene pogreške gomilaju pa čvor se postepeno preko normalnog aktivnog stanja i normalnog pasivnog stanja dovede u stanje zastoja. Kada se čvor nađe u stanju zastoja potrebno ga je ponovno pokrenuti i ponovno mu inicijalizirati početnu konfiguraciju što ga dovodi u stanje spremnosti. Mogući prelasci iz jednog stanja pogreške u drugo stanje pogreške vidljivi su na slici 3.2.




**Sl. 3.2.** Mogući prelasci iz jednog stanja pogreške u drugo.

Kod FlexRay protokola postoji brojač pogrešaka koji se uvećava za 1 svakim parom komunikacijskog ciklusa (parni i neparni ciklus) u kojemu se pojavi barem jedna od dvije pogreške sinkronizacije između FlexRay čvorova koji sudjeluju u komunikaciji. Te pogreške su:

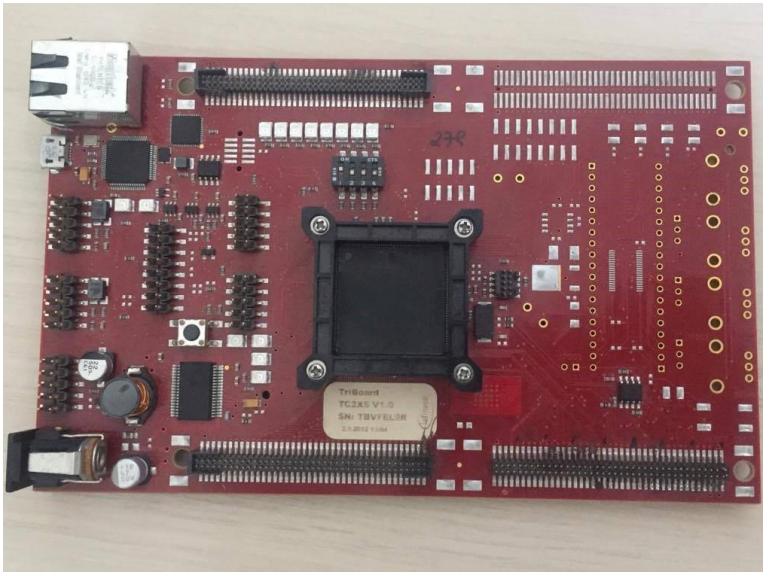
1. Pogreška ispravka brzine prijenosa (engl. *rate correction error*) i
2. Pogreška ispravka odstupanja (engl. *offset correction error*).

Stanje pogreške u kojemu će se nalaziti FlexRay čvor ovisi o brojaču pogrešaka. Prilikom postavljanja početnih vrijednosti, definira se maksimalna vrijednost brojača koju on može poprimiti prije nego što ode u normalno pasivno stanje i maksimalna vrijednost brojača prije nego što čvor pređe u stanje zastoja. Za razliku od CAN protokola gdje se brojač pogrešaka

smanjuje za 1 kada se pošalje ispravna poruka, kod FlexRay protokola brojač pogrešaka resetira se na 0 (početna vrijednost) čim se u potpunosti obavi sinkronizacija.

# 3. PROGRAMSKO RJEŠENJE

U ovome poglavlju bit će prikazano programsko rješenje za oba komunikacijska protokola. Programsko rješenje kod CAN protokola sastoji se od tri različite funkcije koje služe za izazivanje različitih pogrešaka. Te funkcije napravljene su uz pomoć pogonskih programa. Jedna funkcija testira postojeći mehanizam za otklanjanje pogrešaka koji je ugrađen u čip na razvojnoj ploči na način da se simulira neprestano pojavljivanje jedne od mogućih pogrešaka. Druge dvije funkcije služe za izazivanje određenih pogrešaka i prikaz stanja registara zaduženih za praćenje pogrešaka (brojači pogrešaka, upozorenja na pogreške, šifra posljednje pogreške itd.). Programsko rješenje za izazivanje i oporavak od pogrešaka kod FlexRay protokola se sastoji od funkcije koja služi kao rutina oporavka od najtežeg stanja pogreške (stanje zastoja) u kojemu FlexRay čvor može biti (u to stanje pogreške čvor dolazi neprestanim pojavljivanjem bilo koje vrste pogreške). Rutina za oporavak od stanja zastoja napravljena iz razloga što oporavak od tog stanja nije riješen na razini čipa kao kod CAN protokola. Izrađena rutina može se koristiti kao dodatak pogonskim programima. Za izradu programskog rješenja korištena je razvojna ploča Aurix TC 275 (sl. 4.1.) te ploča Aurix TC 297. Od ostalih uređaja korišten je „PCAN-USB: PEAK-System“ (sl. 4.2.) za simulaciju komunikacije zasnovane na CAN protokolu, a za simulaciju FlexRay komunikacije korišten je uređaj „VN 8912“ ot tvrtke Vector Informatik odnosno njegov FlexRay modul „VN 8972“. Programsko rješenje razvijalo se u programu „TASKING TriCore Eclipse v4.3r3“. Osim navedenog programa za razvoj rješenja, korišteni su programi „PCAN-View“ za praćenje CAN prometa te „Canoe 9. 0 SP 5“ za praćenje FlexRay prometa.



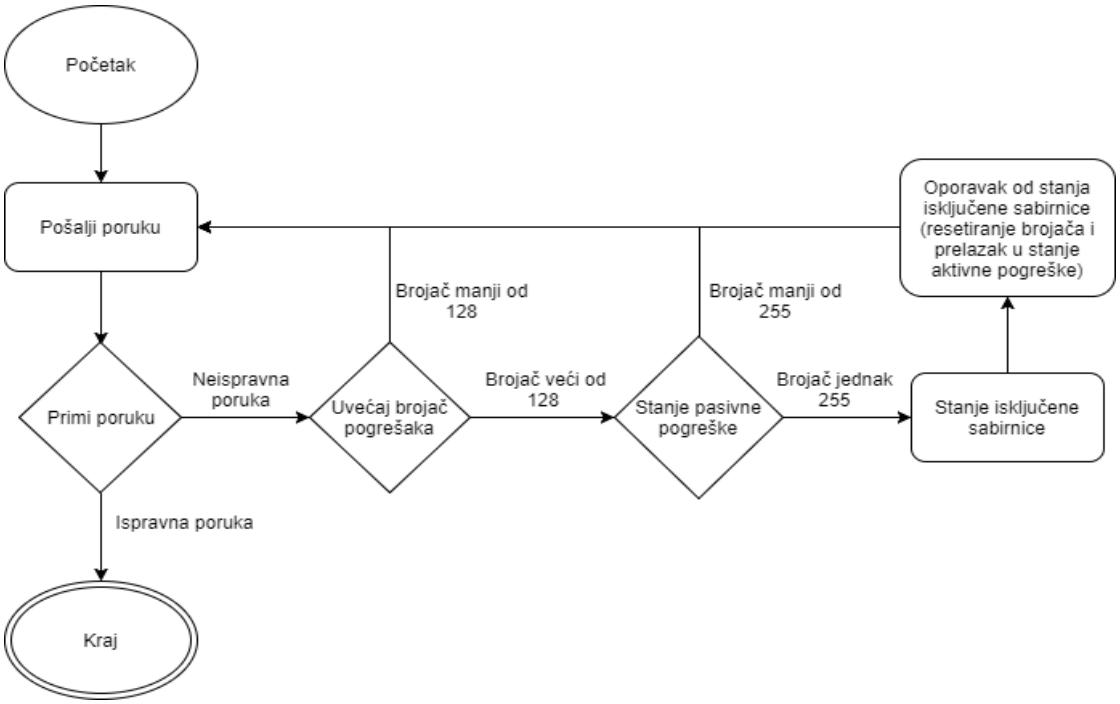
**Sl. 4.1.** Aurix TC 275 razvojna ploča.


**Sl. 4.2.** PCAN-USB: PEAK-System [7].

## 3.1. Programsko rješenje vezano za CAN protokol

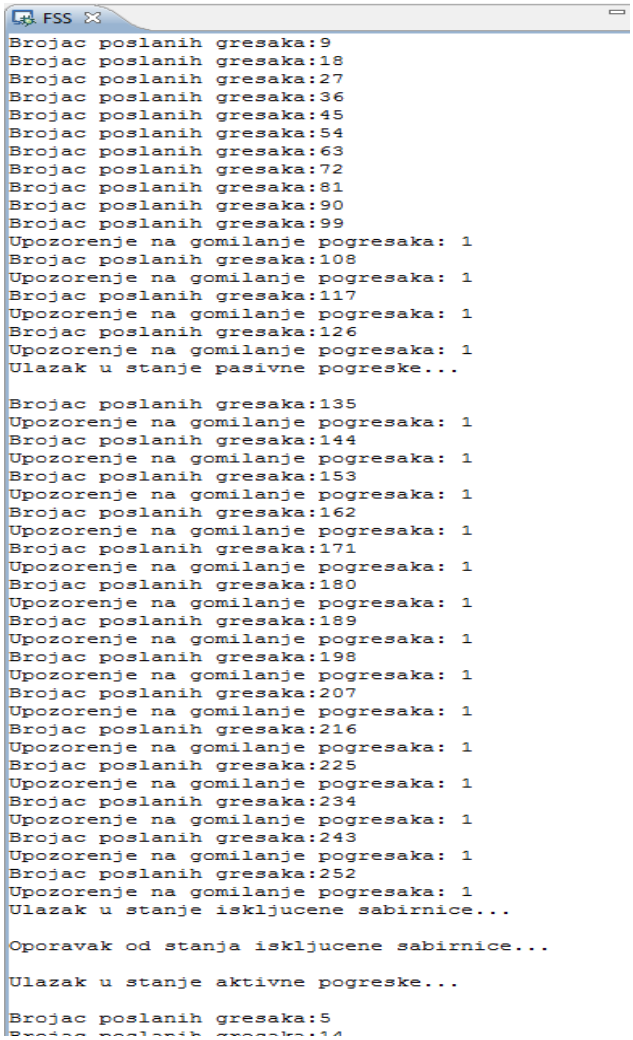
Kao što je u prošlom poglavlju detaljno objašnjeno, CAN protokol razlikuje 5 vrsta različitih pogrešaka od kojih je samo dvije vrste moguće reproducirati softverski, a to su pogreška umetanja dodatnog bita i pogreška potvrde. Za ostale tri vrste pogreške razvojna ploča Aurix TriBoard TC275 ima ugrađena rješenja unutar čipa koja onemogućavaju izazivanje tih pogrešaka od strane programera (to su: pogreška u CRC zaštiti, pogreška forme te pogreška umetanja dodatnog bita). Za simuliranje CAN prometa korišten je (osim čvorova na samoj razvojnoj ploči) i uređaj „PCAN-USB: PEAK-System“ (od sada u tekstu samo PCAN-USB). Za praćenje CAN prometa koristio se PCAN-USB-ov pripadajući program „PCAN-View“.

Prelazak iz jednog stanja pogreške u drugo je također riješeno na razini čipa te se na prelazak i oporavak od stanja pogreške ne može utjecati softverski. Kako bi se to dokazalo napravljena je funkcija (dijagram toka funkcije se nalazi na slici 4.3.) koja simulira slanje poruke s pogreškom pomoću koje CAN čvor može prijeći u sva tri stanja pogreške: aktivna pogreška, pasivna pogreška i stanje isključene sabirnice.




**Sl. 4.3.** Dijagram toka funkcije za simuliranje neprestanog slanja poruke s pogreškom.

Funkcija direktno pristupa registru u kojemu se nalazi brojač poslanih pogrešaka određenog čvora te ga iterativno uvećava za 10 (sl. 4.4.). Brojač poslanih pogrešaka se u normalnim uvjetima uvećava za 8 (kada čvor pošalje neispravnu poruku), ali zbog bržeg i jasnijeg prikaza stanja čvora koristi se vrijednost uvećavanja koja iznosi 10. Funkcija zapravo šalje ispravnu poruku s jednog čvora te na drugom čvoru prima istu (valjanu poruku). Za potrebe prikazivanja promjena stanja čvora, odredišni čvor očekuje različitu poruku od one koju primi (simulira se pojava pogreške tijekom prijenosa). Zbog pojave „pogreške“ pristupa se registru u kojemu se nalazi brojač poslanih pogrešaka te se on uvećava za 10. Na slici 4.4. je vidljiv tijek promjene stanja i uvećavanja brojača, no isto tako vidljivo je da se brojač uvećava za 9.




**Sl. 4.4.** Izgled ispisa funkcije za simuliranje iterativnog slanja poruke s pogreškom.

Razlog tomu je to što se šalju ispravne poruke, a slanjem ispravne poruke brojači pogrešaka se, prema CAN protokolu, smanjuju za 1. Zbog toga se nakon slanja poruke i simuliranja pogreške nad ispravnom porukom prvo brojaču direktno pristupi i uveća ga se za 10, a potom se brojač automatski umanji za 1 zbog prepoznavanja tehnički ispravne poruke. Upozorenje na gomilanje pogrešaka je zastavica (engl. *flag*) koja poprimi vrijednost 1 ako bilo koji brojač pogrešaka (nebitno radi li se o poslanim ili primljenim porukama s pogreškama) prijeđe određenu vrijednost postavljenu u registru razine upozorenja na pogrešku (engl. *error warning level*).

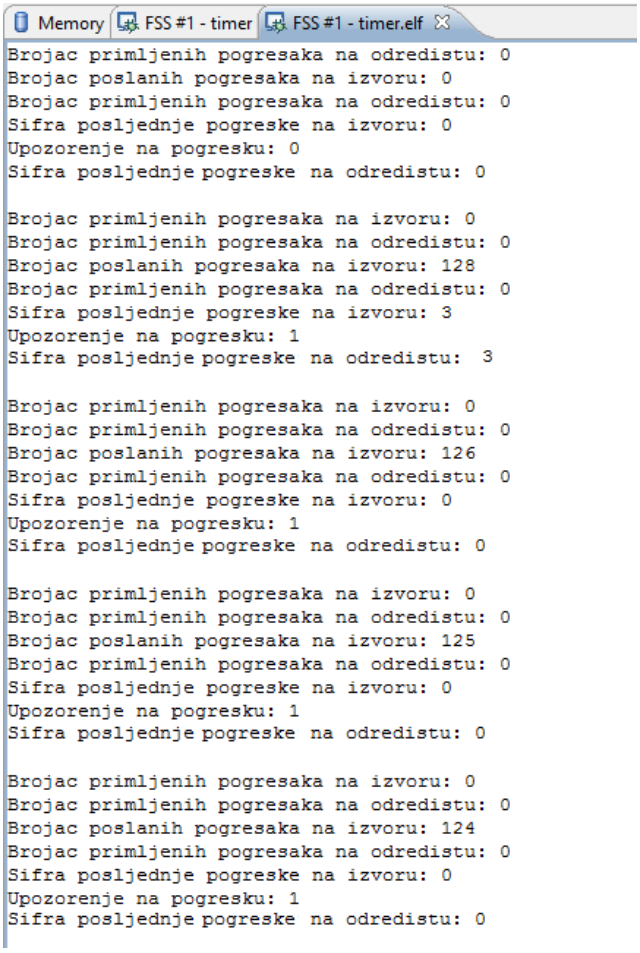
Izazivanje pogreške potvrde napravljen je uz pomoć PCAN-USB uređaj. Pogreška potvrde pojavljuje se kada čvor šalje poruku i ne dobije poruku potvrde od niti jednog čvora u CAN mreži. Za izazivanje te poruke bilo je potrebno napraviti funkciju koja će neprestano slati poruke na PCAN-USB uređaj te koja će pratiti stanje svih brojača pogrešaka i prepoznavati vrstu

pogreške ako do nje dođe. Prepoznati pogrešku koja se posljednja dogodila moguće je uz pomoć pročitane vrijednosti iz registra: šifra posljednje poruke (engl. *Last Error Code*) (Tab 4.1.).

**Tab 4.1.** Moguće vrijednosti šifre posljednje pogreške i njihova značenja.

| **Šifra posljednje pogreške** | **Značenje** |
|---|---|
| 0 | Nema pogreške |
| 1 | Pogreška umetanja dodatnog bita |
| 2 | Pogreška forme |
| 3 | Pogreška potvrde |
| 4 | Pogreška bitova 1 (poslan je recesivni bit, primljen je dominantni) |
| 5 | Pogreška bitova 0 (poslan je dominantni bit, primljen je recesivni) |
| 6 | Pogreška u CRC zaštiti |
| 7 | Jedina vrijednost koju se može ručno upisati u ovaj registar |

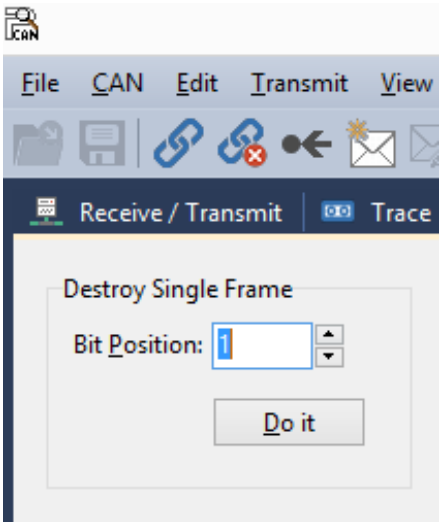
Pogreška potvrde izazvana je tako da se spoje razvojna ploča i PCAN-USB uređaj te se započne slanje poruka od strane razvojne ploče. U jednom trenutku odspoji se PCAN-USB uređaj što uzrokuje pojavljivanje pogreške potvrde, jer je on jedini čvor u CAN mreži koji ima mogućnost primanja poslane poruke. Program u tom trenutku javlja pogrešku potvrde i prikazuje trenutne vrijednosti brojača pogrešaka. Prekid komunikacije na ovoj razvojnoj ploči uzrokuje povećanje vrijednosti brojača poslanih pogrešaka na 128 odnosno na onaj broj s kojim čvor prelazi u stanje pasivne pogreške (sl. 4.5.). Šifra posljednje poruke u ovom slučaju iznosi 3 što potvrđuje pojavljivanje pogreške potvrde. Brojačima pogrešaka na odredištu u ovom slučaju nije moguće pristupiti, jer je odredište PCAN-USB uređaj. Zbog toga se može pratiti samo stanje brojača na izvoru odnosno čvoru koji se nalazi na razvojnoj ploči.




**Sl. 4.5. P**rikaz ispisa funkcije prilikom pojavljivanja pogreške potvrde.

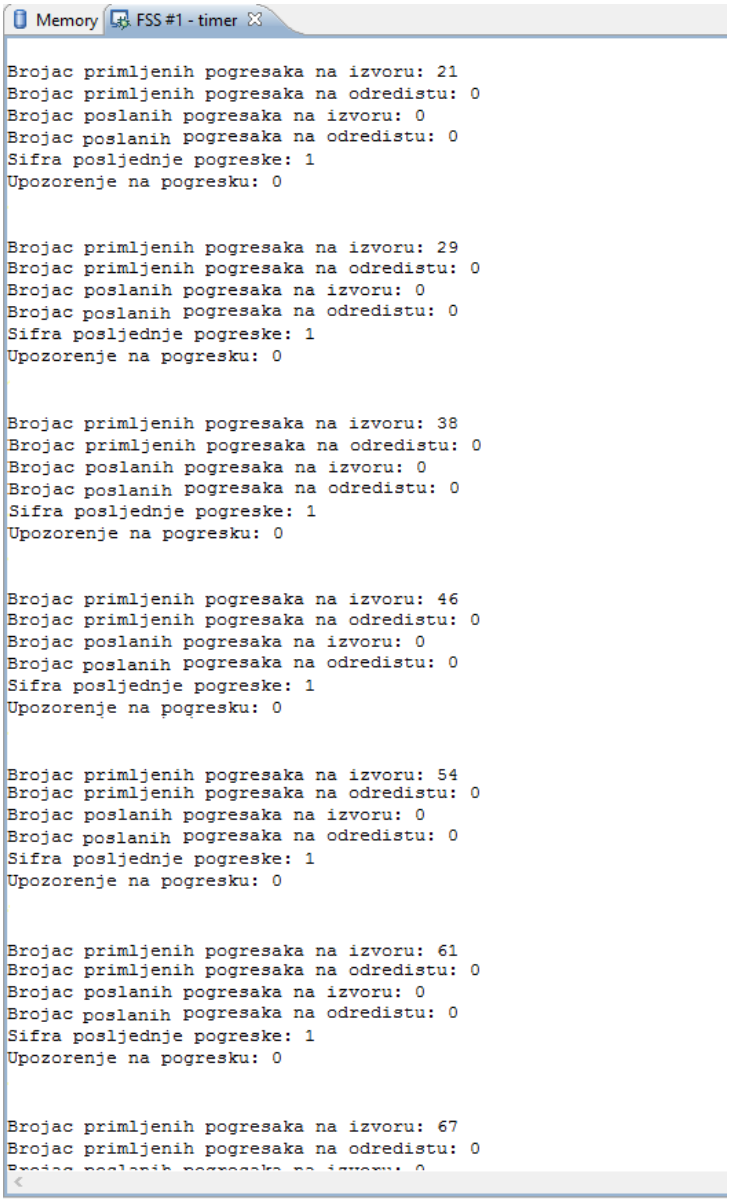
Ponovnim spajanjem razvojne ploče s PCAN-USB uređajem, komunikacija se nastavlja normalnim tokom. Brojač poslanih pogrešaka smanjuje se za 1 za svaku ispravno poslanu poruku. PCAN-USB uređaj je korišten zbog toga što program za praćenje prometa (PCAN-View) na CAN mreži u kojoj se nalazi sam uređaj ima mogućnost prekida veze programski. S tom opcijom nije bilo potrebno fizički odspajati jedan čvor od drugoga.

Druga vrsta izazvane pogreške jest pogreška umetanja dodatnog bita. Kod izazivanja ove pogreške također je korišten PCAN-USB uređaj i program za praćenje njegovog prometa PCAN-View. Program PCAN View ima opciju da namjerno uništi jedan bit u prenesenoj poruci i samim time izazove pogrešku umetanja dodatnog bita (sl. 4.6.) .




**Sl. 4.6.** Izgled sučelja PCAN View programa kojim se uništava pojedini bit poruke kako bi se izazvala pogreška umetanja dodatnog bita.

Za izazivanje ove pogreške korištena je ista funkcija kao i kod izazivanja pogreške potvrde. Funkcija neprestano šalje ispravnu poruku s razvojne ploče na PCAN-USB uređaj, a kada korisnik uništi bit prepoznaje pogrešku i uvećava svoj brojač primljenih poruka (na izvoru). Brojač se uvećava za 1, jer se pogreška događa na PCAN-USB uređaju koji zatim odašilje poruku o pogrešci (pomoću okvira za pogreške) svim čvorovima unutar CAN mreže. Za svaku ispravnu poruku brojač se umanjuje za 1 (sl. 4.7.) . Na slici 4.7. vide se skokovi vrijednosti brojača koji su veći od 1, a razlog tomu je to što je uništeno više bitova zaredom u kratkom vremenskom roku. Program ispisuje vrijednosti 6 različitih brojača i zastavica te zbog trajanja ispisa ne uspijeva prikazati te vrijednosti za svaku pojedinu pogrešku. Razlika između dvije uzastopne vrijednosti brojača primljenih pogrešaka na izvoru prikazuje broj generiranih pogrešaka pomoću uništavanja pojedinog bita. I u ovom slučaju nije moguće pristupiti brojačima pogrešaka na odredištu, jer je odredište PCAN-USB uređaj. Na ovaj način povezivanja razvojne ploče s PCAN-USB uređajem moguće je samo pratiti promet na CAN mreži. Od ostalih ispisa vrijednosti prikazana je šifra poruke koja u ovom slučaju ima vrijednost 1. Prema tablici 4.1. vidljivo je da šifra 1 označava pojavu pogreške umetanja dodatnog bita. Ispis vrijednosti registra koji upozorava na pogrešku ima vrijednost 0 sve dok se ne pređe vrijednost od 98 bilo kojeg brojača pogrešaka.

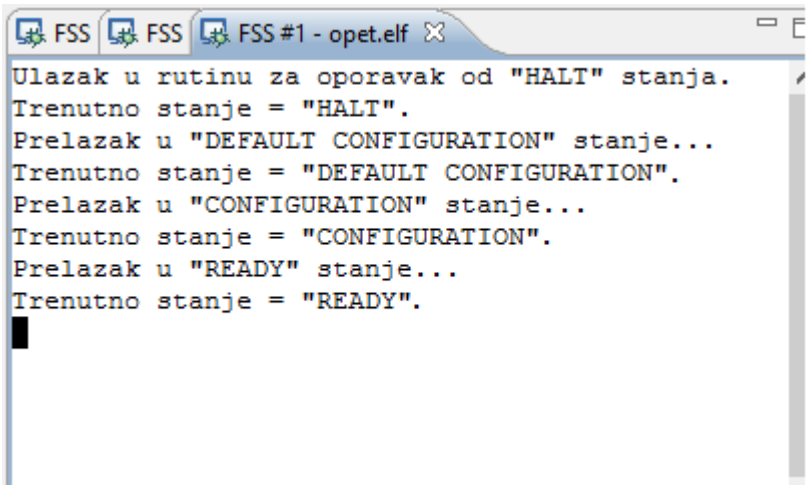



**Sl. 4.7.** Izgled ispisa rezultata funkcije za slanje poruka uz namjerno izazivanje pogreške umetanja dodatnog bita.

Oporavak od stanja pogreške u CAN protokolu je riješen u potpunosti pomoću programa koji se već nalaze na čipu razvojne ploče. To se vidi iz nemogućnosti izazivanja pogrešaka bez pomoći PCAN-USB uređaja, ali i iz simulacije slanja poruke s pogreškom između CAN čvorova na razvojnoj ploči. Testiranje CAN protokola odrađeno je uspješno i bez većih poteškoća.

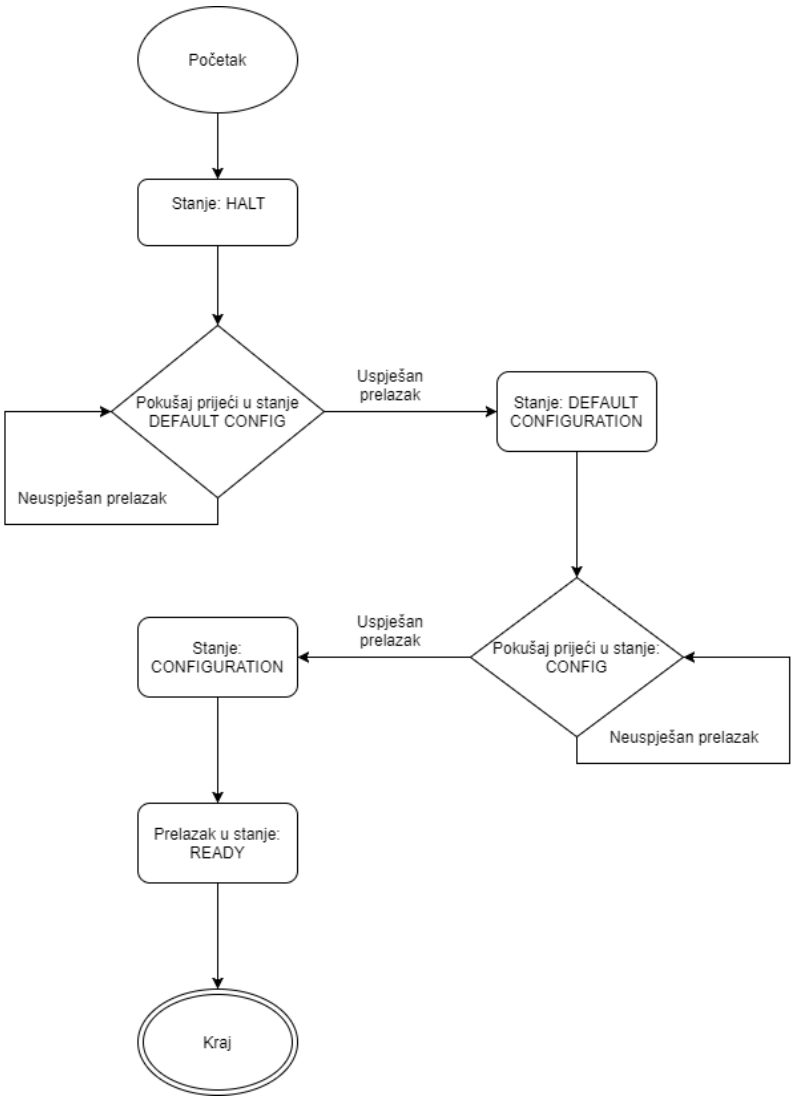
## 3.2. Programsko rješenje vezano za FlexRay protokol

Najčešće pogreške u komunikaciji koja se zasniva na FlexRay protokolu su pogreške sinkronizacije. U prošlom poglavlju navedena su dva tipa pogrešaka sinkronizacije: pogreška sinkronizacije brzine prijenosa i pogreška odstupanja. Osim ta dva tipa pogrešaka, navedena su i tri stanja pogreške u kojima se FlexRay čvor može nalaziti: normalno aktivno stanje, normalno pasivno stanje te stanje zastoja. Zbog ograničenosti razvojne ploče i njenih pogonskih programa (više o toj temi u sljedećem potpoglavlju) nije bilo jednostavno ispitati oporavak od pogrešaka FlexRay protokola. Kako je ovaj dio rada prilagođen mogućnostima razvojne ploče, nije bilo moguće izazvati specifične poruke i pratiti stanje brojača pogreške kao kod CAN protokola. No, i bez mogućnosti izazivanja pogrešaka pronađen je način da se FlexRay čvor dovede u sva tri stanja pogreške. Prilikom spajanja dva FlexRay čvora nije ostvarena sinkronizacija između istih i iz tog razloga FlexRay čvor nakon početnog postavljanja konfiguracije odlazi u stanje zastoja. To je stanje pogreške u kojemu se onemogućava slanje i primanje poruka i ono predstavlja najgoru situaciju u kojoj se jedan FlexRay čvor može naći. Oporavak od ovog stanja pogreške nije riješen na razini ugrađenog čipa razvojne ploče. Da bi se FlexRay čvor oporavio od stanja zastoja, potrebno ga je ponovno konfigurirati i nakon toga dovesti u stanje spremnosti iz kojega može prijeći u normalno aktivno stanje pogreške. Funkcija oporavka poziva se u slučaju da FlexRay čvor pređe u stanje zastoja. Iz stanja zastoja slijedno se prelazi u stanje predefinirane konfiguracije potom u stanje konfiguracije i na kraju u stanje spremnosti (sl. 4.8.).




**Sl. 4.8.** Ispis stanja kroz koja funkcija za oporavak od stanja zastoja prolazi prije nego dođe do oporavka.

Iz stanja spremnosti FlexRay čvor može prijeći u normalno aktivno stanje pogreške i tako nastaviti neometanu komunikaciju. Osim pomoću ove funkcije (sl. 4.9.), iz stanja zastoja moguće je izaći (oporaviti se) tako da se FlexRay čvor ugasi i ponovno upali.




**Sl. 4.9.** Dijagram toka funkcije za oporavak od stanja zastoja.

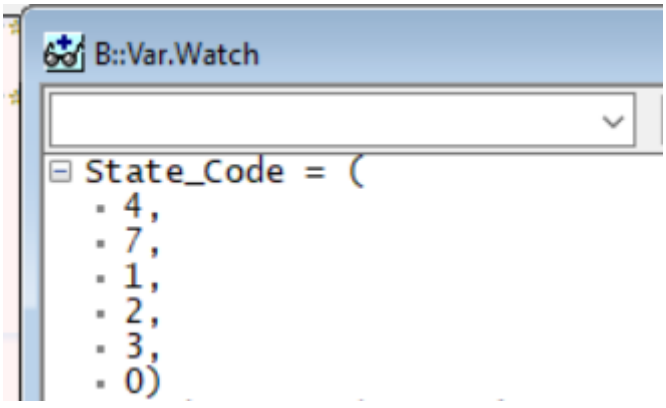
Zbog ranije spomenutih razloga testiranje funkcije za oporavak od stanja zastoja u komunikaciji između FlexRay čvorova nije bilo izvedivo na razvojnoj ploči koja je predviđena za izvođenje ovog diplomskog rada. Kako bi se testirala funkcija oporavka u FlexRay mreži u kojoj je omogućena komunikacija između čvorova, koristila se razvojna ploča Aurix TC297. Na toj razvojnoj ploči nalazi se funkcionalan FlexRay modul koji je zahtijevao minimalne preinake u kodu kako bi funkcija radila. Prilikom testiranja korišten je i puno moćniji razbubnik (engl. *debugger*) Lauterbach i njegova aplikacija Trace32 s kojom se lagano prati tijek izvođenja

programa i stanje u registrima. Jedino ograničenje korištenja ove ploče bilo je to što nije bio omogućen pristup serijskoj konzoli (engl. *console*) na kojoj bi se vidio ispis rezultata. Iz tog razloga napravljeno je polje cjelobrojnih vrijednosti u koje se zapisuju šifre pojedinog stanja pogreške (Tab. 4.2.), ako u to stanje pogreške FlexRay čvor uđe.

**Tab. 4.2.** Šifre stanja pogrešaka korištene za praćenje stanja u kojima se čvor nalazi i njihovo značenje.

| **Šifra stanja** | **Značenje šifre** |
|---|---|
| 1 | Stanje predefinirane konfiguracije (engl. *default configuration*-DEFAULTCONFIG) |
| 2 | Stanje konfiguracije (engl. *configuration*-CONFIGURATION) |
| 3 | Stanje spremnosti (engl. *ready*-READY) |
| 4 | Normalno aktivno stanje |
| 5 | Normalno pasivno stanje |
| 7 | Stanje zastoja (engl. *halt*-HALT) |

Šifra stanja u kojoj se FlexRay čvor trenutno nalazi zapisan je na prvo slobodno mjesto unutar polja cjelobrojnih vrijednosti na taj način vidi se tijek promjena stanja (sl. 4.10.) i bez ispisa kakvog je funkcija imala na prvobitnoj razvojnoj ploči.




**Sl. 4.10.** Prikaz vrijednosti polja u koje se zapisuje šifra stanja u kojoj se trenutno FlexRay čvor nalazi.

Izazivanje pogreške ostvareno je fizičkim odspajanjem sabirnice pomoću koje je veza ostvarena nakon što je FlexRay čvor doveden u normalno aktivno stanje pogreške. Iz slike 4.7. vidljiv je tijek prelazaka stanja od normalnog aktivnog stanja do stanja spremnosti. Zbog toga što je jedini način da se izazove pogreška bilo odspajanje sabirnice nakon kojeg se vrlo brzo FlexRay čvor prebaci u stanje zastoja nije bilo moguće pratiti brojač pogrešaka.

## 3.3. Problemi tijekom rada

Izazivanje i oporavak od pogrešaka kod FlexRay protokola nije bilo jednostavno kao kod CAN protokola. Naime, veći dio ovog dijela rada svodio se na osposobljavanje pogonskih programa FlexRay protokola za razvojnu ploču. Pogonski programi koji su priloženi uz ploču nisu funkcionirali na ispravan način, a detaljan opis svih funkcija i korištenih varijabli ne postoji. Pokretanjem pogonskih programa došlo je do greške u izvršavanju koda. Nakon iscrpnog traženja greške otkriveno je da postoji problem s okidanjem funkcija prekida (engl. *interrupt*). Kako bi se taj problem popravio, bez promjene funkcija pogonskih programa, bilo je potrebno izravno pristupiti registrima i omogućiti okidanje funkcija prekida. Nakon rješavanja problema s okidanjem funkcija prekida i potpunog osposobljavanja pogonskih programa pojavio se novi problem kod sinkronizacije FlexRay čvorova. Naime, predefinirane postavke pogonskih programa nisu riješile sinkronizaciju između FlexRay čvorova.

# 4. ZAKLJUČAK

Zadatak ovog diplomskog rada bio je detaljnije upoznavanje rukovanja greškama u dva najčešće korištena protokola u automobilskoj industriji, a to su CAN i FlexRay. Rad se koncentrirao na izazivanje pogrešaka prilikom komunikacije i provjeru oporavljanja od istih. Nakon upoznavanja s razvojnim okruženjem započelo se s pokušavanjem izazivanja pogrešaka prilikom komunikacije koja se zasniva na CAN protokolu. Uz pomoć PCAN-USB uređaja izazvane su neke od mogućih pogrešaka te je pokazano da se CAN čvorovi samostalno oporavljaju od najtežih stanja pogreške. Taj oporavak temelji se na ugrađenim funkcijama u čipu razvojne ploče što je dokazano uz pomoć korištenja pogonskih programa. Oporavak od pogreške kod CAN protokola radi dobro i nije se morao dorađivati, a izazivanje pogrešaka bez korištenja PCAN-USB uređaja bilo je neuspješno i zbog toga je zaključeno da je rukovanje s greškama veoma dobro napravljeno. Kod FlexRay protokola pojavili su se prvi veći problemi. Dosta napora je bilo uloženo u osposobljavanje pogonskih programa Nakon otklanjanja poteškoća s pogonskim programima napravljeno je rješenje za oporavak od stanja zastoja FlexRay čvora. Rješenja su u potpunosti funkcionalna za oba protokola. Za razliku od CAN protokola, FlexRay nije imao samostalno rješenje za oporavak od takvog stanja. Pogreške koje nije bilo moguće izazvati tijekom komunikacije zasnovane na FlexRay protokolu dovele bi, u najgorem slučaju, FlexRay čvor u stanje zastoja za čiji oporavak je napravljeno programsko rješenje. Zbog toga nije od velike važnosti koje pogreške su izazvane, a koje nisu, jer sve rezultiraju istom promjenom stanja pogreške.

# LITERATURA


[1] „Introduction to CAN“. [Na internetu]. Dostupno na: https://elearning.vector.com/vl_can_introduction_en.html. [Pristupljeno: 04-ruj-2017].

[2] „Introduction to the CAN.pdf“. [Na internetu]. Dostupno na: http://www.ti.com/lit/an/sloa101b/sloa101b.pdf. [Pristupljeno: 04-ruj-2017].

[3] „Introduction to FlexRay“. [Na internetu]. Dostupno na: https://elearning.vector.com/vl_flexray_introduction_en.html. [Pristupljeno: 04-ruj-2017].

[4] „FlexRay-EN.pdf“. [Na internetu]. Dostupno na: http://www.fujitsu.com/downloads/CN/fmc/lsi/FlexRay-EN.pdf. [Pristupljeno: 04-ruj-2017].

[5] „FlexRay Automotive Communication Bus Overview-National Instruments“. [Na internetu]. Dostupno na: http://www.ni.com/white-paper/3352/en/. [Pristupljeno: 04-ruj-2017].

[6] „FlexRayCommunicationSystem.pdf“. [Na internetu]. Dostupno na: http://www.softwareresearch.net/fileadmin/src/docs/teaching/SS08/PS_VS/FlexRayCommunicationSystem.pdf. [Pristupljeno: 04-ruj-2017].

[7] „PCAN-USB pro FD: PEAK System“. [Na internetu]. Dostupno na:http://www.peak-system.com/PCAN-USB-Pro-FD.366.0.html?&L=1 [Pristupljeno: 04-ruj-2017].

## SAŽETAK

Funkcionalnosti današnjeg automobila ovise o radu i komunikaciji između kontrolnih jedinica. Pogreške koje se mogu pojaviti prilikom komunikacije između tih kontrolnih jedinica trebaju biti na vrijeme prepoznate te ih se treba riješiti u što kraćem roku. Ovaj rad bavi se izazivanjem pogrešaka i oporavkom od istih kod sustava koji koriste CAN i FlexRay komunikacijske protokole. CAN protokol koristi se za manje kritične sustave zbog manje brzine i zbog njegove komunikacije pogonjene događajima. U drugu ruku, FlexRay protokol koristi se u najkritičnijim sustavima automobila zbog svoje velike brzine i komunikacije pogonjene vremenskim okidačima. U sklopu ovog rada ispitana je mogućnost izazivanja i oporavka od pogrešaka tijekom komunikacije s navedenim protokolima na razvojnoj ploči Aurix TC 275. Uspješno su simulirane pogreške koje je moguće proizvesti za pojedini protokol te je napravljena funkcija za oporavak od pogrešaka što nije riješeno na razini čipa na korištenoj platformi.



Ključne riječi: automobil, komunikacija, pogreška, oporavak, CAN, FlexRay, protokol.

## ABSTRACT

The functionality of today's car is dependent on work and communication between control units. Errors that may occur during communication between these control units should be timely recognized and resolved in the shortest possible time. This paper deals with causing errors and recovering from the same in systems using CAN and FlexRay communication protocols. The CAN protocol is used for less critical systems due to lower speed and its event-driven communication. On the other hand, the FlexRay protocol is used in the most critical car systems due to its high speed and time-triggered based communication. Within this work, the ability to cause and recover from error during the communication with the mentioned protocols on the Aurix TC 275 development board is tested. Successful simulations of errors were produced for each protocol and a fault recovery function was written for the fault recovery that is not resolved at chip level.

Keywords: car, communication, error, recovery, CAN, FlexRay, protocol.

# ŽIVOTOPIS

Josip Turjak rođen je 10.12.1993. u Osijeku. Svoje školovanje započeo je upisom u osnovnu školu Mladost u rujnu 2000.-te godine. Po završetku osnovne škole, u rujnu 2008. upisuje III. gimnaziju Osijek koju završava u lipnju 2012.-te godine. U rujnu iste godine upisao je sveučilišni preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku. Završetkom preddiplomskog sveučilišnog studija računarstva u rujnu 2015. godine stekao je zvanje: prvostupnik (baccalaureus) inženjer računarstva (univ. bacc. ing. comp.). Svoje obrazovanje nastavio je u rujnu 2015. godine upisom na sveučilišni diplomski studij računarstva, smjer programsko inženjerstvo na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Obranom ovog diplomskog rada završit će navedeni studij i steći zvanje magistar inženjer računarstva (mag. ing. comp). Trenutno odrađuje praksu u tvrtki RT-RK u kojoj će se po završetku studija i zaposliti. Od stranih jezika ima znanje B2 razine engleskog i A1 razine njemačkog jezika. Od računalnih vještina vrlo dobro poznaje C, C++ i C# programske jezike te je upoznat s programskim jezikom java i jezicima za razvoj internet tehnologija kao što su html 5, css i JavaScript.