UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
Odsek za elektrotehniku i računarstvo
Institut za računarstvo i automatiku
Katedra za računarsku tehniku i
računarske komunikacije

Implementacija V5.1 sprege upotrebom programskog jezgra za realizaciju konačnih automata

Diplomski rad

MENTOR:
Prof.dr. Miroslav Popović

STUDENT:
E5508 Laslo Benarik

Novi Sad, Jun 2003.

S a d r ž a j

1. Uvod

U okviru ovog diplomskog rada trebalo je izvršiti transformaciju postojećeg programskog rešenja V5.1 sprege u novo programsko rešenje zasnovano na programskom jezgru za realizaciju konačnih automata.

Sem toga trebalo je obezbediti sledeće funkcije:

- paralelan rad više V5.1 sprega u okviru jednog sistema,
- povezivanje V5.1 sprege sa elementima fizičke arhitekture
- dodavanje i brisanje varijanti koje služe za konfigurisanje pojedinačnih V5.1 sprega,
- startovanje, zaustavljanje i brisanje V5.1 sprega
- povezivanje sa delom za nadzor i upravljanje.

Polazna osnova u realizaciji ovog programskog rešenja je ranije realizovana V5.1 sprega u skladu sa preporukama i standardima G.964 od strane ITU (koji se poziva na preporuke Q.920 i Q.921) odnosno ETS 300 324-1 od strane ETS (koji se poziva na preporuku ETS 300 125) i programsko jezgro za realizaciju konačnih automata koje je razijeno na Katedri za računarsku tehniku i računarske komunikacije.

Realizacija ovog rada pre svega je zahtevala upoznavanje sa navedenim preporukama i standardima, upoznavanje i razumevanje dobijenog programskog koda i upoznavanje programskog jezgra za realizaciju konačnih automata.

Programski kod koji predstavlja polaznu osnovu za realizaciju V5.1 sprege se oslanja na staro programsko jezgro u kojem su programski resursi: upravljanje memorijom, upravljanje porukama i vremenske kontrole, realizovani kao globalne promenljive a ceo sistem automata funkcioniše samo u okviru jedne programske niti.

Kod novog programskog jezgra resursi sistema su smešteni u poseban objekat koji je instanca klase FSMSystem. Ovaj objekat upravlja ovim resursima a pristup istim je moguć samo kroz objekte koji su instance klasa koje predstavljaju konačne automate u ovom sistemu, a celokupno programsko jezgro je realizovano tako da podržava rad sa više programskih niti.

Nakon izvršene transformacije programskog rešenja V5.1 sprege bilo je neophodno izvršiti proveru ispravnosti dobijenog programskog rešenja.

Verifikaciju realizovanog programskog rešenja je trebalo obaviti pomoću skupa test slučajeva definisanih u okviru preporuka ETS 300 324-4 i ETS 300 324-8. U ovim preporukama dat je detaljan opis svih potrebnih struktura podataka i procedura za implementaciju dva specijalna programa V51DLL i V51NWKAN koji služe za testiranje programskih rešenja V5.1 sprege a za koje se želi potvrda o saglasnosti sa preporukom ETS 300 324-1.

Kako su navedeni programi V51DLL i V51NWKAN razvijeni na Katedri za računarsku tehniku i računarske komunikacije bilo je potrebno iskoristiti ove programe sa njihovim celokupnim skupom ulaznih parametara kako bi se testirale sve grane programskog koda koji predstavlja V5.1 spregu.

Provera realizovanog programskog rešenja je trebalo da se obavi na modelu sastavljenom od dva računara međusobno povezani E1 linkom. Svaki od računara bi morao da sadrži odgovarajuću karticu kako bi se E1 link mogao priključiti.

Jedan računar bi vršio funkciju LE strane V5.1 sprege, dok bi drugi , gde bi bilo aktivirano realizovano programsko rešenje vršio funkciju AN strane V5.1 sprege.

Nakon postavljenog modela za testiranje bilo je potrebno potrvrditi uspešnost svih test slučajeva definisanih u standardima.

Sem toga trebalo je proširiti postojeće programsko rešenje s delom koda za istovremeno postojanje više V5.1 sprega te definisati skup testnih slučajeva za proveru ispravnosti realizovanog koda.

Bilo je neophodno realizovati osnovni skup komandi dela za nadzor i upravljanje.

2. V5.1 sprega

2.1. V5.1 standard

Savremeni telekomunikacioni sistemi zbog raznolikosti krajnjih korisnika i prenosnih puteva ne dozvoljavaju jednostavno a pre svega jeftino priključenje na mrežu. U odsustvu međunarodnih standarda za povezivanje na digitalnu pristupnu mrežu došlo se do situacije u kojoj su pojedini proizvođači ili pak zemlje razvili posebne standarde za priključenje krajnjih korisnika na mrežu.

Kako bi se izbegla masa tih standarda definisan je poseban standard V5 kojim se nezavisno od signalizacije u lokalnoj centrali definiše veza između pristupne mreže AN (AN – Access Network) i lokalne centrale LE (LE – Local Exchange). Rad na V5 standardima započet je sredinom 1991. godine od strane Evropskog instituta za telekomunikacione standarde (ETSI – European Telecommunications Standards Institute).

U periodu od 17 meseci specijalna grupa eksperata definisala je dva koncepta V5 sprege, prvi baziran na principu statičkog multipleksera pod nazivom V5.1 sprega i drugi baziran na dinamičkom principu koncentratora pod nazivom V5.2 sprega, pri čemu je omogućena evolucija sa V5.1 na V5.2 spregu.

V5.1 sprega je skup električnih, fizičkih, proceduralnih i protokolarnih zahteva čije ispunjenje obezbeđuje programsku podršku za povezivanje pristupne mreže AN sa lokalnom centralom LE. Pristup mreži u opštem slučaju ukazuje na sve ono što je potrebno da bi se krajnji korisnici priključili na mrežu.

Električni i fizički zahtevi V5.1 sprege zasnovani su na karakteristikama jednog linka (E1) od 2048 kbit/s unutar koga se podaci prenose u 64 kbit/s kanalima a koji je definisan u preporukama ETS 300 166 i ETS 300 167.

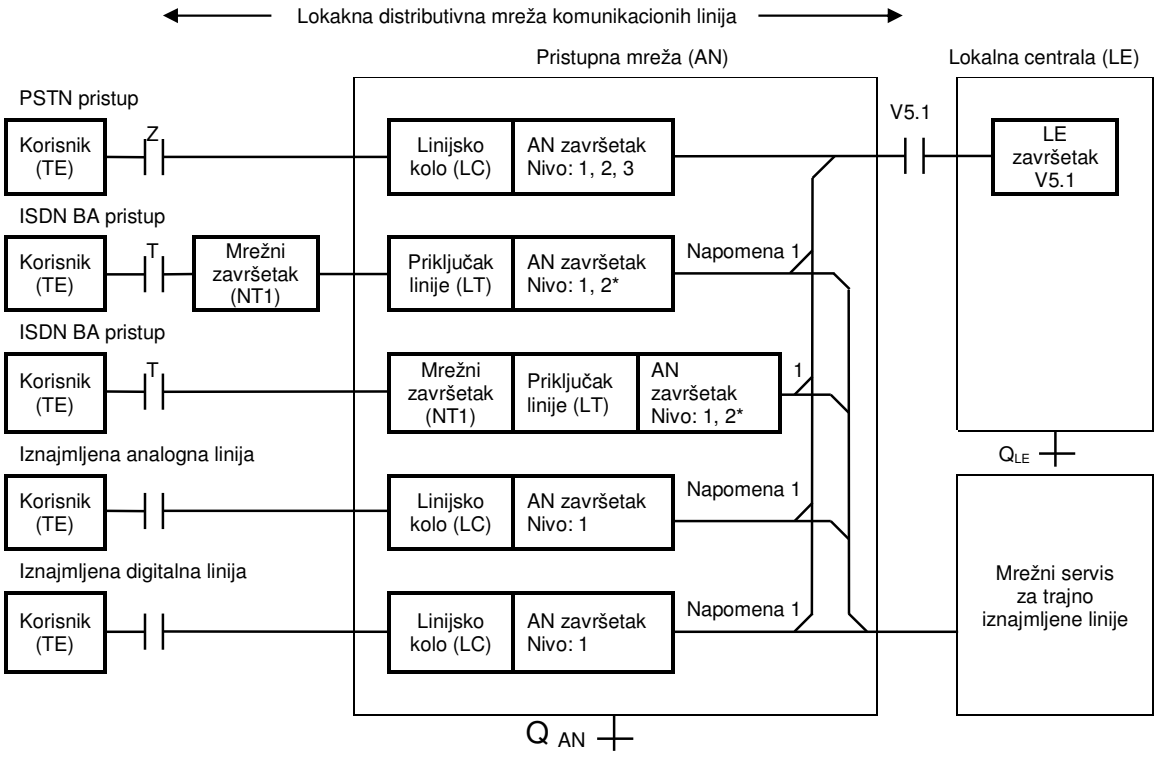
Proceduralni i protokolarni zahtevi omogućuju da veza na strani pristupa mreži podržava priključenje digitalnih i analognih korisnika. Ovo je ostvareno kroz dve grupe servisa.

Prvu grupu čine servisi koji ostvaruju vezu na zahev korisnika i to su:

- **PSTN** (PSTN – Public Switched Telephone Network) protokol podržava pojedinačne korisnike i analogne PABX (PABX – Private Automatic Brach eXchange) komutacije, obe sa ili bez DTMF (DTMF – Dual Tone Multiple Frequency) i sa ili bez dodatnih usluga dok za PABX komutacije postoji i dodatna mogućnost sa ili bez DDI (DDI – Direct Dialling In). V5.1 dozvoljava maksimalno 30 PSTN korisnika.

- **ISDN BA** (ISDN BA – Integrated Services Digital Network Basic Access). podržava priključenje digitalnih korisnika na S ili T referentnu tačku kao i ISDN PABX komutacije na NT2 'T' referentnu tačku. Kako je V5.1 sprega transparentna za ISDN korisnike, nema nikakvih ograničenja na usluge koje se pružaju ISDN korisnicima. V5.1 dozvoljava maksimalno 15 ISDN BA korisnika.

Drugu grupu servisa čini podrška za trajne veze upotrebom jednog ili oba B kanala od ISDN BA priključka. Trajne veze se mogu podeliti na stalne ili polu-stalne iznajmljene linije. Trajne veze omogućuju analognu ili digitalnu komunikaciju bez signalizacije po drugim kanalima.



Slika 1. Prikaz servisa koje pruža V5.1 sprega sa AN strane

Napomena 1: Izbor kanala i pridruživanje servisa su deo konfigurisanja V5.1 Sprege. Zvezdica označava da je drugi nivo samo delimično zatvoren unutar AN strane.

2.2. Struktura protokol steka V5.1 sprege

OSI (OSI – Open Systems Interconnection) referentni model koji je definisan u standardu ISO 7498 predstavlja osnovu strukture protokol steka V5.1 sprege. Ovaj model definiše slojevitu hijerarhijsku arhitekturu koja omogućuje da se složeni zadaci iz oblasti komunikacija dekomponuju na module sa kojima se lakše upravlja.

Nivoi u ovom modelu komuniciraju primitivama i to tako da samo susedni nivoi međusobno razmenjuju poruke. Na slici 2. prikazana je hijerarhija OSI referentnog modela i paralelno sa njim hijerarhija nivoa V5.1 sprege koja se preslikava na samo prva tri nivoa OSI referentnog modela.

	7 Nivoa OSI modela		V5.1 Nivoi
7	Nivo Aplikacije		
6	Nivo Prezentacije		
5	Nivo Sesije		
4	Transportni nivo		
3	Mrežni nivo		ETS 300 324-1, Uspostava poziva
2	Nivo kanala		ETS 300 125, LAPV
1	Fizički nivo		ETS 300 166 i ETS 300 167

Slika 2. OSI referentni model i V5.1 sprega

2.3. Arhitektura V5.1 Sprege

Funkcionalni opis V5.1 sprege prikazan je na slici 3.

Slika 3. Funkcionalni opis V5.1 sprege

- **Prenosni kanali** (Bearer channels): služe za dvosmernu komunikaciju kroz ISDN B-kanale za ISDN BA korisnički pristup ili kroz PCM (PCM – Pulse Code Modulation) kodirane 64 kbit/s kanale za PSTN korisnički pristup.

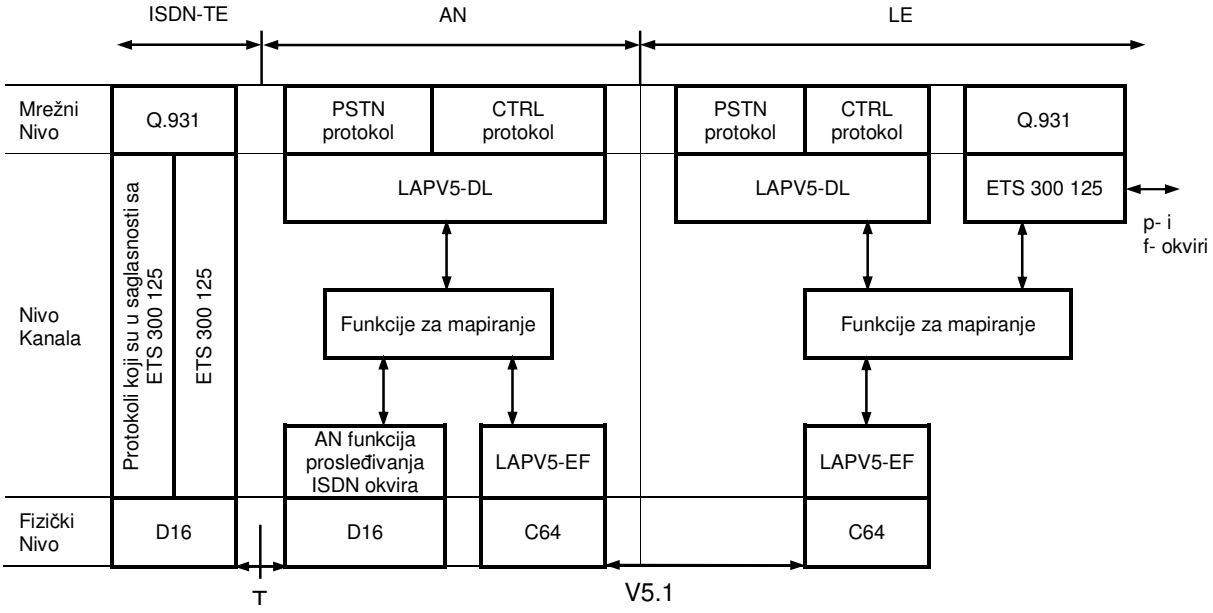
- **ISDN signalizacija** (ISDN D channel information): je dvosmerna razmena informacija za D-kanal kod ISDN BA korisničkog pristupa (uključujući tu i Ds-, p- i f- tipove poruka).

- **PSTN signalizacija** (PSTN signaling information): je dvosmerna razmena informacija za signalizaciju PSTN korisničkih pristupa.

- **Upravljačka signalizacija** (Control): sadrži u sebi informacije o nizu aktivnosti koje su neophodne za upravljanje V5.1 spregom. U ovu grupu spadaju:
  - upravljanje korisničkim pristupom obezbeđuje dvosmernu razmenu informacija o statusu i komandama za svaki pojedinačni korisnički priključak.
  - upravljanje 2048 kbit/s linkom što podrazumeva poravnavanje grupe ili pojedinačnih I okvira za prenos podataka, indikaciju alarma i kontrolne sume za 2048 kbit/s.
  - upravljanje kanalima drugog nivoa koji omogućuju dvosmernu komunikaciju za prenos signalizacije PSTN i CTRL protokola (CTRL – Control protokol).
  - podršku za funkcije koje na sinhronizovani način izvršavaju rekonfiguraciju i restartovanje V5.1 sprege.

- **Vremenske kontrole** (Timing): služi da pruži neophodne informacije o vremenskim kontrolama koje se koriste za prenos bita, pravilnu identifikaciju okteta i sinhronizaciju okvira za prenos podataka. Ove informacije takođe mogu biti upotrebljene i za sinhronizaciju aktivnosti između pristupne mreže AN i lokalne centrale LE.

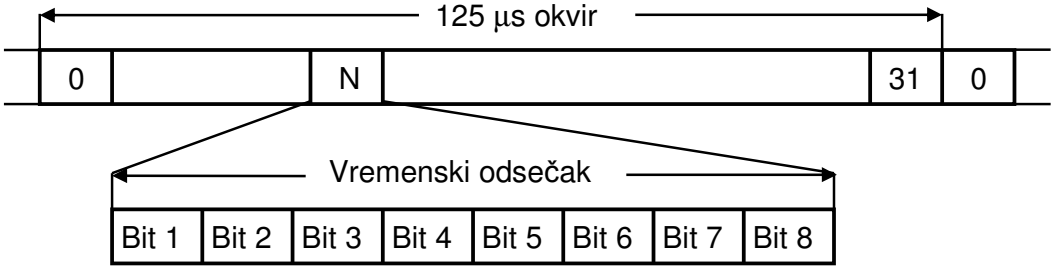
V5.1 sprega sastoji se iz više različitih protokola koji su sastavni delovi drugog i trećeg nivoa. Naredna slika prikazuje strukturu protokola u okviru V5.1 sprege.



Slika 4. Arhitektura protokola V5.1 sprege

Fizički Nivo (Physical Layer): čini standardni E1 link koji se sastoji od 32 vremenska odsečka (C64). Detaljan opis E1 linka je dat u okviru ETS 300 166 i ETS 300 167 preporukama. Odsečak 0 je rezervisan za poravnanje okvira, a preostalih 31 se koriste za prenos signalnih poruka i korisničkih informacija.

U okviru V5.1 sprege za komunikacioni kanal se može rezervisati više od jednog vremenskog odsečka. Vremenski odsečci koji se koriste za prenos korisničkih informacija nazivaju se kanali nosioci (Bearer Time Slot). Brzina prenosa po odsečku je 64 kbit/s.



Slika 5. Format standardnog E1 linka

V5.1 sprega prilikom konfiguracije zahteva jedan na jedan mapiranje kanala nosioca i korisničkih priključaka. Promena rasporeda može se obaviti posebnom procedurom CTRL protokola. Zbog toga se kaže da je za V5.1 spregu dodela kanala statička.

Svakom komunikacionom protokolu dodeljuje se po jedan komunikacioni put (C-kanal) za prenos signalnih poruka. Dodela se obavlja po sledećim pravilima:

- minimalno mora postojati jedan C-kanal a maksimalno tri C-kanala,
- mapiranje C-kanala kod V5.1 sprege na određene vremenske osečke se obavlja u fazi konfiguracije V5.1 sprege i to:

- C-kanal 1 – vremenski odsečak 16,
- C-kanal 2 – vremenski odsečak 15,
- C-kanal 3 – vremenski odsečak 31,

- za C-kanal 1 se uvek vezuje informacioni tok podataka CTRL protokola, dok se signalizacija PSTN protokola kod PSTN korisničkog pristupa i informacioni tok podataka ISDN D-kanala kod ISDN BA korisničkog pristupa mogu dodeliti bilo kom od navedena tri C-kanala.

Nivo kanala (Data Link Layer): predstavlja protokol LAPV5 (LAPV5 – Link Access Protocol for V5 interface) ovaj protokol je detaljno opisan u ETS 300 125. U okviru ovog nivoa vrši se multipleksiranje različitih informacionih tokova u komunikacione kanale.

LAPV5 ima zadatak da formira i šalje okvire sa neophodnom sinhronizacijom, kontrolom grešaka i kontrolama za upravljanje tokom podataka. Sadrži u sebi funkcije za aktiviranje, održavanje i deaktiviranje kanala.

U slučaju V5.1 sprege drugi nivo se sastoji iz četiri dela i to su:

- **LAPV5-DL** (LAPV5 data link sublayer), podnivo linka podataka služi za pouzdan prenos poruka protokola trećeg nivoa njihovim parnim protokolima.

- **LAPV5-EF** (LAPV5 envelope function sublayer), podnivo za prenos paketa omogućava prenos paketa između AN i LE.

- **LAPV5-FR** (AN frame relay function), AN funkcija prosleđivanja ISDN okvira. Ova funkcija izvršava: multipleksiranje i demultipleksiranje poruka na osnovu adrese korisničkog priključka, ograničavanje i provera ispravnosti okvira.

- **Funkcije mapiranja** (Mapping function), predstavlja modul koji je zadužen za komunikaciju predhodna tri podnivoa protokola nivoa kanala.

Mrežni nivo (Network Layer): PSTN signalizacija za PSTN korisničke priključke se multipleksira na ovom nivou i prenosi se kroz jedan kanal na drugom nivou. Adrese i informacije koje se odnose na pojedine korisničke priključke sadržane su u porukama trećeg nivoa PSTN i CTRL protokola. Treći nivo se sastoji iz sledećih protokola:

- **PSTN protokol** – je protokol trećeg nivoa (mrežni nivo) koji obezbeđuje prenošenje stanja na analognom vodu putem poruka po zajedničkom signalnom kanalu. Većina stanja u kojima može da se nađe analogni vod se ne interpretiraju od strane protokola nego se samo transportuju poruke od programskog modula pretvarača linijskih (hook-on i hook-off) i registarskih (digit) signala analognog korisničkog porta na AN strani do nacionalnog protokola kontrole poziva na LE strani i obrnuto. U preporuci se daje obiman skup signalnih informacionih elemenata koji bi trebao da bude dovoljan za sve danas postojeće nacionalne protokole.

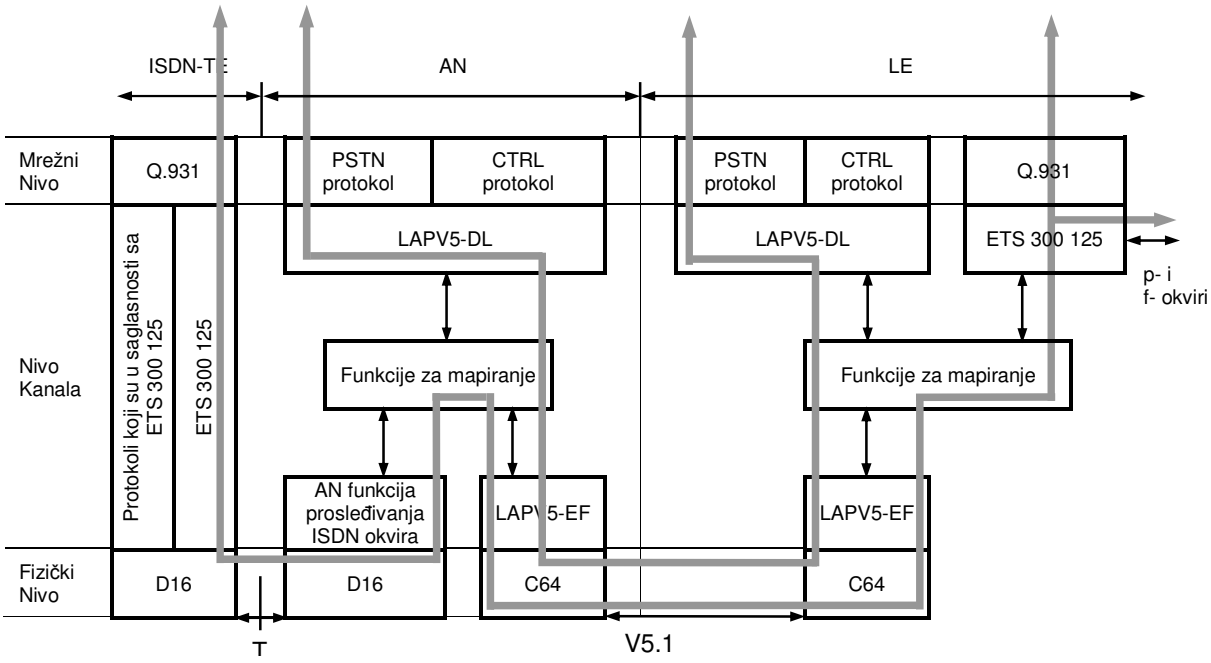
- **CTRL protokol** – je protokol trećeg nivoa koji integriše kontrole korisničkih priključaka i podršku za rekonfiguraciju (reprovisioning) V5.1 sprege. Postoje dve vrste kontrole korisničkih priključaka: za ISDN BA (osnovni pristup) i za PSTN. Za svaki od njih je potrebno obezbediti praćenje statusa i mogućnost kontrolisanog prevođenja iz blokiranog u neblokirano stanje i obrnuto u slučaju otkaza ili potrebe za održavanjem.

Upravljanje V5.1 spregom (System Management): predstavlja poseban skup automata, funkcija i procedura koji služe kao sprega za komunikaciju između svih navedenih protokola i funkcija unutar V5.1 sprege međusobno tako i sa korisnicima koji se brinu o funkcionisanju celokupnog sistema.

Upravljanje ne predstavlja poseban protokol već integriše sve nabrojane protokole u jedinstven sistem. Formalni zahtevi u vezi sa upravljanjem V5.1 sprege se svode na procedure startovanja, zaustavljanja i ponovnog startovanja sistema u kojima mora da se ispoštuje logički redosled uključivanja pojedinih automata. Rukovanje korisničkim priključcima kao i promena konfiguracije celokupnog sistema u toku njegovog rada su takođe u nadležnosti ovog dela V5.1 sprege.

U opštem slučaju upravljanje V5.1 spregom se vrši preko TMN-a (TMN – Telecommunication Management Network) kroz Q_{AN} spregu koja je definisana u preporukama ETS 300 376-1 kao što se može videti na slici 1.

Na slici 6. prikazani su informacioni tokovi za PSTN i ISDN signalizaciju.



Slika 6. PSTN i ISDN Signalizacioni tokovi

3. Programsko jezgro za realizaciju konačnih automata

3.1. Elementi programskog jezgra

Komunikacioni protokoli se najjednostavnije mogu realizovati kao skup konačnih determinističkih automata koji međusobno razmenjuju poruke koje su definisane datim protokolom. Da bi se olakšala implementacija komunikacionih protokola uvedene su klase koje predstavljaju mehanizme automata a koji su nezavisni od problema koji rešava konkretni automat.

Programsko jezgro je sastavljeno iz sledećeg niza funkcionalnih mogućnosti koje se koriste u realizaciji komunikacionih protokola:

- Sistemske rutine jezgra dostupne aplikaciji,
- Sadrži osnovnu klasu za realizovanje konačnih automata, a koja obezbedjuje sledeće funkcionalnosti:
  - Praćenje stanja,
  - Definisanje funkcija prelaza,
  - Odredjivanje funkcije prelaza u zavisnosti od stanja automata i koda poruke,
  - Rukovanje porukama,
- Izvršava razmenu poruka izmedju automata,
- Obezbeđuje pravilno funkcionisanje skupa automata,
- Rukovanje memorijom, zauzimanje i oslobadjanje bafera za poruke.
- Rukovanje vremenskim kontrolama.

Primenom ovakvog programskog jezgra rešavanje konkretnog problema se svodi na rastavljanje datog komunikacionog protokola na celine koje se mogu predstavlti kao konačni automati, zatim se za svaki automat nasledi osnovna klasa za realizaciju konačnih automata i u okviru ove nove klase kodira se zadata funkcionalnost preko funkcija prelaza konačnog automata.

3.2. Staro programsko jezgro

Programski kod koji predstavlja osnovno programsko rešenje V5.1 sprege i koji je iskorišćen za razvoj programskog rešenja koje je prikazano u ovom radu je zasnovano na starom programskom jezgru.

Osnovne karakteristike ovog programskog jezgra su:

- Osnovni resursi jezgra kao što su ograničeni delovi memorije, sistem za upravljanje vremenskim kontrolama i sistem za razmenu poruka su definisani kao globalne promenljive tj. ovi resusi su dostupni iz bilo kog dela programskog koda.
- Osnovna klasa koja predstavlja konačni automat ima samo najosnovnije elemente koje definišu automat. Ukoliko je potrebno da postoji više istih automata u sistemu formira se jedna instanca date klase koja u sebi sadrži nizove struktura koje su potrebne za svaki automat.
- Sistem za razmenu poruka samo skladišti poruke dok je svaki automat odgovoran da sam proveri da li ima poruka za njega, posledica ovog pristupa je da svaki automat moze da uzme poruku namenjenu bilo kom drugom automatu.

3.3. Novo programsko jezgro

U okviru ovog programskog jezgra postoji mnogo klasa od kojih za definisanje autoamata treba izdvojiti dve i to klasu FiniteStateMachine i klasu FSMSystem.

Klasa FiniteStateMachine je dizajnirana tako da u potpunosti enkapsulira funkcije koje su neophodne za uspesno funkcionisanje bilo kog konačnog automata. Ovakav dizajn omogućuje da programeri prilikom implementacije programskih rešenja usmere svu svoju pažnju na funkcionalnost automata koje implementiraju kroz implementaciju funkcija prelaza.

Prilikom inicijalizacije svakog automata poziva se niz funkcija u okviru kojih se definišu funkcije prelaza i vremenske kontrole u sledećem obliku:

- **void InitEventProc**(STANJE, PORUKA, (Adresa)&FUNKCIJA_OBRADE),
- **void InitUnexpectedEventProc**(STANJE, (Adresa)&FUNKCIJA_OBRADE),
- **void InitTimerBlock**(NAZIV_TIMERA, VREDNOST_BROJAČA, PORUKA),

Funkcijom InitEventProc se formira tabela prelaza automata. Funkcijom InitUnexpectedEventProc se definiše funkcija obrade za nedefinisane poruke. Dok se InitTimerBlock funkcijom se inicijalizuju vremenske kontrole koje će koristiti ovaj automat.

Osnovne funkcije programskog jezgra koje automati koriste su:

- Funkcije za rukovanje memorijom:
  - **uint8 *GetBuffer**(uint32 dužina),
  - **void RetBuffer**(uint8 *bafer),
- Funkcije za rukovanje vremenskim kontrolama:
  - **uint8 *StartTimer**(NAZIV_TIMERA),
  - **void StopTimer**(NAZIV_TIMERA),
  - **bool IsTimerRunning**(NAZIV_TIMERA),
- Funkcije za rukovanje porukama:
  - **void PrepareNewMessage**(0x00, NAZIV_PORUKE),
  - **void SetMsgToAutomate**(AUTOMAT_KOJI_PRIMA_PORUKU),
  - **void SetMsgObjectNumberTo**(id_Automata_koji_prima_poruku),
  - **void SendMessage**(RED_ZA_ULANCAVANJE_PORUKA),

Za razliku od starog programskog jezgra gde se potreba da u sistemu funkcioniše više automata istog tipa rešava strukturama podataka unutar jedne instance klase datog automata kod novog programskog jezgra se ovaj problem prosto rešava objektnim pristupom tj. formira onaj broj instanci date klase automata koliko je potrebno za rešavanje datog zadatka.

Kada se implementiraju sve klase koje predstavjaju konačne automate koji su potrebni za rešavanje datog problema, sve automate potrebno je povezati u jedan sistem koji će funkcionisati kao celina. Ovo povezivanje se ostvaruje preko klase FSMSystem.

Ova klasa sadrži osnovne resurse za rukovanje memorijom, rukovanje vremenskim kontrolama i vodi računa o rukovanju porukama. Klasa FSMSystem se koristi tako sto se instancira jedan objekat ove klase preko koje se vrši inicijalizacija programskog jezgra na osnovu definisanih parametara kao što su su veličina i broj bafera. Zatim se prijavljuju sve instance automata koje treba da funkcionišu u sistemu i u posebnoj programskoj niti se poziva funkcija start().

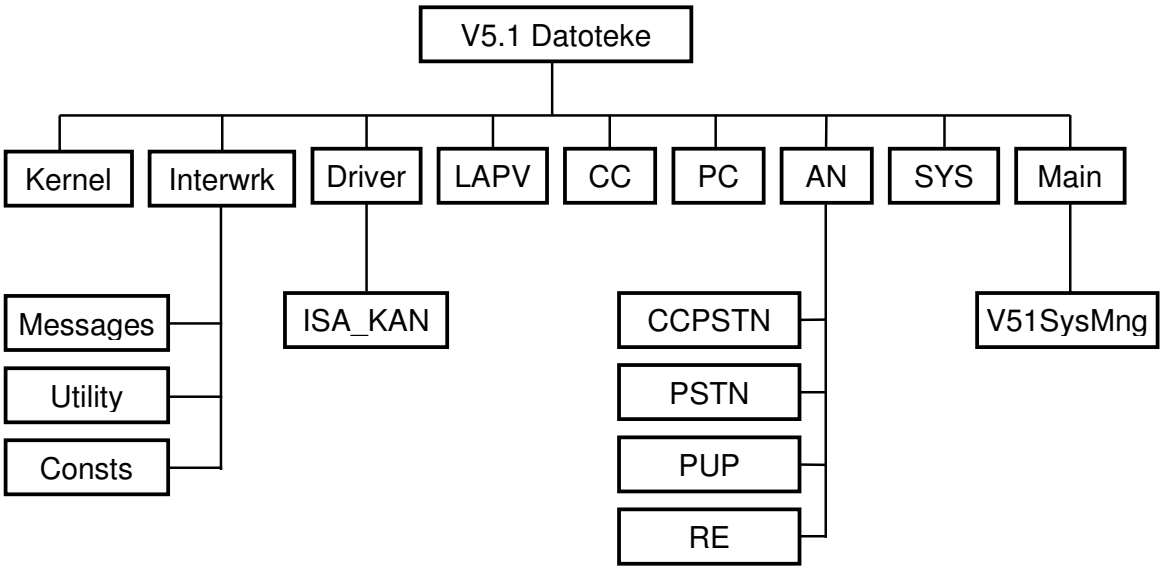
Ova funkcija u beskonačnoj petlji prosleđuje sve poruke odgovarajućim automatima i poziva funkcije prelaza na osnovu informacija koje se nalaze u poruci i tabele prelaza automata koji je primio poruku.

4. Realizacija programskog rešenja V5.1 sprege

4.1. Struktura programskog rešenja V5.1 sprege

Programske datoteke koje ulaze u sastav programskog rešenja su raspoređene unutar hijerarhiske strukture stabla tako da grane ovog stabla sačinjavaju manje celine koje rešavaju deo problema a sve datoteke zajedno sačinjavaju programsko rešenje V5.1 sprege.

Ovakva struktura je proizašla iz visokog nivoa složenosti tehničke dokumentacije standarda V5.1 sprege koja oslikava funkcionalnost koju podržava V5.1 sprega. Zahtev da programsko rešenje podržava paralelan i nezavisan rad više V5.1 sprega čine problem utoliko složenijim. Na slici 7. dat je prikaz ove strukture.



Slika 7. Struktura programskog rešenja V5.1 sprege

- **Kernel**: sadrži datoteke novog programskog jezgra,
- **Interwrk**: ovde su grupisane sve osnovne i pomoćne klase koje su potrebne za realizaciju ovog programskog rešenja, sadrži podgrane:
  - **Messages**: sadrži datoteke u kojima su definisane klase koje reprezentuju poruke drugog i trećeg nivoa V5.1 sprege,
  - **Utility**: grupiše klase koje predstavljaju pomoćne strukture podataka koje se koriste u drugim klasama,
  - **Consts**: sadrži datoteke sa vrednostima koje su konstantne za ceo projekat,
- **Driver**-**ISA_KAN**: sadrži programsku podršku za fizički nivo V5.1 sprege,
- **LAPV**: se sastoji od datoteka u kojima su definisane klase drugog nivoa,
- **CC** i **PC**: sadrže klase u kojima je definisan CTRL protokol trećeg nivoa
- **AN**: sa svojim podgranama sadrži klase koje definišu PSTN protokol trećeg nivoa kao i automate za rukovanje PSTN korisničkim priključcima.
- **SYS**: sadrži skup klasa koje realizuju upravljački podsistem V5.1 sprege,
- **Main** sa **V51SysMng**: sadrži programske datoteke u kojima se nalazi kod za pravilnu inicijalizaciju celokupnog sistema kao i klase koje služe za rukovanje i upravljanje sa više instanci V5.1 sprege.

4.2. Osnovne i pomoćne klase
4.2.1. Klasa V5FiniteStateMachine

Klasa FiniteStateMachine je klasa koja je definisana u okviru programskog jezgra i ona sadrži u sebi sve što je potrebno da bi jedna klasa predstavljala automat koji može pravilno da funkcioniše u sistemu automata. Sa druge strane svi automati u ovom programskom rešenju imaju neke zajedničke osobine i karakteristike koje je iz praktičnih razloga poželjno smestiti u jednu baznu klasu koju bi posle svi automati nasledili po potrebi.

Klasa V5FiniteStateMachine nasleđuje pomenutu klasu programskog jezgra i sadrži u sebi atribute i objekte koji su zajednički za sve automate u ovom programskom rešenju. Iz ovih razloga klasa V5FiniteStateMachine je osnovna klasa za realizaciju svih konačnih automata u okviru ovog programskog rešenja V5.1 sprege.

Programski kod ove klase smešten je u datoteku V5fsm.cpp sa zaglavljem u V5fsm.h. Atributi i Metode klase V5FiniteStateMachine mogu se podeliti u tri grupe gde svaka grupa ima posebnu funkciju.

U prvu grupu spadaju virtuelne funkcije iz klase FiniteStateMachine koje moraju da se definišu u svakom automatu. Ove funkcije koristi programsko jezgro da bi se pravilno rukovalo porukama automata u sistemu automata. Pošto su ove funkcije identične za sve automate u ovom projektu, definisane su unutar ove klase i to su:

- MessageInterface *GetMessageInterface(uint32 id),
- void SetDefaultHeader(uint8 infoCoding),
- void SetDefaultFSMData(),
- void NoFreeInstances().

Zbog ovih metoda neophodno je bilo dodati i dva atributa koji predstavljaju instance klasa za rukovanje porukama:

- V5MsgCoding v5MsgCoding,
- StandardMessage StandardMsgCoding.

Ove funkcije i atributi zajedno sa ostalim funkcijama programskog jezgra obezbeđuju automatsko rukovanje zaglavljem svake poruke odnosno pojednostavljuje upotrebu i rukovanje slanjem i prijemom poruka a po prijemu i poziv odgovarajuće funkcije obrade kod automata kome je poruka poslata.

Drugu grupu čine atributi i funkcije koji su zajednički za sve ostale automate u ovom projektu, oni se dalje mogu podeliti:

- Na atribute koji čuvaju informacije o konfiguraciji V5.1 sprege:
  - **cVariantConfig *m_ptrVariantCfg**, je pokazivač na instancu klase cVariantConfig koja sadrži u sebi informacije o tome kako je konfigurisana instanca V5.1 sprege kojoj ovaj automat pripada,
  - **CCPathMapping *m_ptrCPathMapp**, je pokazivač na objekat u kome su smeštene informacije o mapiranju komunikacionih kanala.
- Na atribut i funkciju koji su potrebni za rukovanje instancama V5.1 sprege:
  - **V51InterfaceNode *m_ptrV51**, ovaj atribut sadrži pokazivač na instancu V5.1 sprege kojoj ovaj atribut pripada,
  - **uint32 GetObjectId**(), preko ove funkcije se dobija jedinstveni identifikacioni broj automata koji svaki automat ima u sistemu automata.

- I na atribute i funkcje za rukovanje porukama V5.1 sprege pošto ove poruke predstavljaju samo informacioni deo poruka koje se razmenjuju između automata u okviru sistema automata.
  - **L3Message m_L3msg**, je atribut koji sadrži u sebi funkcije za rukovanje elementima koji su smešteni unutar poruke,
  - **uint8 *GetParamV5Msg**(), funkcija izdvaja V5.1 poruku iz informacionog dela poruke koja se razmenjuje među automatima.
  - **uint8 *AddParamV5Msg**(L3Message message), funkcija dodaje V5.1 poruku u informacioni deo poruke koja se razmenjuje među automatima,
  - **void SendMsgTo**(uint8 fsmID, uint32 objectID, uint8 mbxID), ovom funkcijom se pojednostavljuje slanje pruka pošto se sa ovom jednom funkcijom koja ima tri argumenta, koji tačno identifikuju adresu na koju data poruka treba da stigne, izbegava kodiranje niza funkcija svaki put kada je potrebno da se pošalje poruka.

Trećoj grupi pripadaju atributi i funkcije koje služe za testiranje kako pojedinih automata tako i celog sistema u celini. Ovi elementi su posebno izdvojeni i ograničeni predprocesorskim direktivama (#ifdef __V5_FSM_TEST__ i #endif) tako da se ovaj kod može vrlo jednostavno izostaviti kada se završi testiranje. Najosnovniji elemeti ove grupe su:

- **static FileClass *file**, je jedinstvena instanca koja enkapsulira datoteku u koju se upisuju sve aktivnosti svih automata u toku njihovog rada kada se sprovodi testiranje automata,
- **void Process**(uint8 *msg), je redefinisana funkcija klase FiniteStateMachine**,** koju programsko jezgro poziva prilikom obrade poruke tj. poziva funkciju obrade u skladu sa tabelom prelaza datog automata, kako bi mogle da se zabeleže dijagnostičke informacije o radu automata i celog sistema,
- **void MsgInfo**(), formira zapis stanja i osnovnih informacija koje se mogu dobiti iz primljene poruke u datoteku koja je definisana za skupljanje ovih informacija,
- **char* FSMType**(uint8 type), funkcija vraća naziv automata u zavisnosti od vrednosti argumenta type,
- **virtual char *MsgCodeName**(uint16 codeName), ovo je virtuelna funkcija koja ima za zadatak da vrati naziv poruke odgovarajućeg automata za zadati argument codeName. Funkcija je virtuelna kako bi bilo moguće da svaki automat redefiniše ovu funkciju i time omogući ovoj klasi da u datoteku upiše stvarni naziv poruke koji je automat primio.

4.2.2. Klasa V5MsgCoding

Ova klasa takođe spada u grupu osnovnih klasa zato što je kao i klasa V5FiniteStateMachine u vrlo tesnoj vezi sa programskim jezgrom. Rukovanje porukama u sistemu automata je jedan od poslova programskog jezgra pri čemu se programsko jezgro oslanja na specijalne klase u kojima su definisane funkcije za rukovanje informacionim poljima unutar poruke. Programski kod ove klase smešten je u datoteku V5MsgCoding.cpp sa zaglavljem u V5MsgCoding.h

Bazna klasa za obradu poruka u programskom jezgru je MessageInterface a kako je ovo virtuelna klasa ona mora biti nasleđena a njene virualne metode implementirane u novoj klasi.

U okviru programskog jezgra postoji klasa koja implementira klasu MessageInterface i to je klasa StandardMessage. Ova klasa je dodata programskom jezgru kao bi postojala klasa koja može na jednostavan i standardan način da rukuje sa porukama.

Jedna od osobina klase StandardMessage je da je za dužinu poruke, koja može da se pravilno obrađuje ovom klasom, odeđen jedan bajt što samu poruku ograničava na 256 bajtova. Kako je najveća dužina poruke koja je definisana V5.1 spregom 533 bajtova, ova osobina klase StandardMessage postaje ograničenje za ovaj projekat.

Zbog navedenog razloga razvijena je nova klasa V5MsgCoding koja takođe nasleđuje klasu MessageInterface ali za dužinu poruke ima rezervisana 2 bajta što ograničava dužinu poruke na 64kbajta.

U klasi V5MsgCoding su definisane funkcije analogno funkcijama u klasi StandardMessage stim da su sve funkcije promenjene kako bi se pravilno rukovalo informacionim poljima u poruci. Novi raspored informacionih polja u novoj klasi izgleda:

- oktet 0: kod parametra, naziv informacionog dela poruke,
- oktet 1:
- oktet 2: dužina informacionog dela poruke (oktet 1 + oktet 2),
- oktet 3: prvi bajt informacionog dela poruke

4.2.3. Klase VariantConfig i CCPathMapping

V5.1 sprega prilikom svoje inicijalizacije na osnovu unapred definisane konfiguracije formira odgovarajuće strukture podataka i aktivira odgovarajuće automate. Ista procedura se događa kada se posebnim funkcijama CTRL protokola iz trećeg nivoa aktivira proces promene konfiguracije.

Kod V5.1 sprege ova konfiguracija je statička, to podrazumeva da su sve moguće konfiguracije kojima može da se pokrene V5.1 sprega unapred definisane i poznate na AN i LE strani. Promena konfiguracije je moguća i svodi se samo na izbor jedne iz skupa onih koje su definisane.

Klase VariantConfig i CCPathMapping su definisane tako da objekti koji su instance ovih klasa predstavljaju trenutnu konfiguraciju V5.1 sprege. Ovi objekti su jedinstveni za jednu V5.1 spregu. Programski kod ovih klasa smešten je u datotekama Config.cpp i Cpathmap.cpp sa zaglavljima u Config.h i Cpathmap.h.

Klasa VariantConfig je definisana tako da sadrži opšte informacije o konfiguraciji V5.1 sprege. Objekat ove klase inicijalizuje se skupom podataka koji predstavlja jednu konfiguraciju (provisioned data set). Iz ovog skupa podataka funkcijama koje su definisane u ovoj klasi izvlače se sledeće informacije:

- Podaci koje definišu V5.1 spregu:
  - **Variant**, je broj varijante koja je upotrebljena za konfiguraciju V5.1 sprege. Ovaj broj je jedinstven za svaku pojedinačnu konfiguraciju koja je pridružena jednoj V5.1 sprezi.
  - **Interface ID**, jedinstveni identifikacioni broj V5.1 sprege, upotrebljava se za identifikovanje V5.1 sprege u sistemima gde više njih radi paralelno.

- **Originating call prevail**, prvenstvo za prosleđivanje poziva ukoliko korisnik u istom trenutku želi da uspostavi poziv i treba da primi poziv.
- Informacije koje definišu tipove i raspored korisničkih priključaka na pojedine vremeske odsečke,
- Raspored komunikacionih kanala koji se koriste za signalizaciju.

S druge strane klasa CCPathMapping sadrži specifične informacije o konfiguraciji komunikacionih puteva koji se koriste za signalizaciju u V5.1 sprezi. Unutar ove klase smeštene su informacije o mapiranju komunikacionih puteva na odgovarajuće komunikacione kanale koji su definisani u konfiguraciji V51. sprege a da su pri tome usaglašeni sa standardom za V5.1 spregu.

Ova klasa se inicijalizuje na osnovu informacija iz klase VariantConfig a kasnije se samo koristi kao tabela u kojoj su mapirane sve putanje.

4.2.4. Klase L3Message i L2Message

V5.1 standard definiše kako se nizovi bajtova grupišu tako da formiraju okvire, koji predstavljaju poruke koje parni procesi koriste za međusobnu komunikaciju. Pošto su ove poruke definisane tako da se sastoje od zaglavlja poruke i njenog informacionog dela formirane su dve klase L3Message i L2Message koje imaju funkcije za rukovanje pojedinim delovima odgovarajuće poruke.

Instance klasa L3Message se koriste za obradu poruka trećeg nivoa dok se instance klase L2Message koriste za obradu poruka drugog nivoa. Kako se poruke sa viših hijerarhiskih nivoa u protokolima prenose u okviru informacionog dela poruka sa nižih informacionih nivoa u ovom programskom rešenju ove klase su implementirane tako da je klasa L3Message bazna klasa a da ovu klasu nasleđuje klasa L2Message.

Programski kod ovih klasa smešten je u datotekama L3msg.cpp i L2msg.cpp sa zaglavljima u L3msg.h i L2msg.h.

Najosnovniji elementi klase L3Message su:

- Atribut **uint8 *Msg**, je pokazivač na prvi bajt u nizu koji predstavlja poruku,
- I sledeće funkcije:
  - **void Create**(uint8 *buffer), ovom funkcijom se niz bajtova za koji se zna da formatiran kao poruka trećeg nivoa pridružuje instanci ove klase kako bi se na jednostavan način rukovalo sadržajem poruke.
  - **void Create**(uint8 *buffer, uint8 signalId, uint16 portId), služi za formiranje nove poruke koja je definisana argumentom signalId u datom baferu pri čemu je u argumentu portId data vrednost adrese PSTN korisničkog priključka.
  - **void CreateISDN**(uint8 *buffer, uint8 signalId, uint16 portId), služi za formiranje nove poruke koja je definisana argumentom signalId u datom baferu pri čemu je u argumentu portId data vrednost adrese ISDN korisničkog priključka.
  - **uint8 GetMsgId**(), funkcija vraća vrednost koji je tip poruke,
  - **uint16 GetUserPortId**(), funkcija vraća vrednost korisničkog priključka,
  - **uint8 *GetMsgBuffer**(), funkcija vraća adresu memorijske lokacije prvog bajta iz poruke,
  - **uint8 *GetParameter**(uint8 code), funkcija vraća adresu parametra u okviru poruke koji je tražena preko argumenta code.

- **uint8* AddParam**(uint8 *param, uint8 *newBuffer), funkcija kopira staru poruku u novi bafer pri čemu joj dodaje parametar dat argumentom param i vraća adresu starog bafera kako bi se taj bafer vratio programskom jezgru.

Pošto se poruke u obe klase posmatraju kao nizovi bajtova, klasa L2Message nema novih atributa već sadrži samo nove funkcije za rukovanje poljima koje poruke drugog nivoa poseduju. Najvažnije funkcije su:

- **uint16 GetEFAddr**() i void SetEFAddr(), za pristup polju EFaddres ,
- **uint16 GetV5DLAddr**() i void SetV5DLAddr(uint16 addr), za pristup polju V5DLaddress
- **uint8 GetCRBit**() i void SetCRBit(uint8 bitValue), za obradu polja koji označava da li je poruka komanda ili odgovor (command/response bit),
- **uint8 GetMsgType**() i void SetMsgType(uint8 type), za pristup polju koje definiše tip poruke,
- **uint8 GetNR**() i void SetNR(uint8 nr), za pristup NR polju,
- **uint8 GetNS**() i void SetNS(uint8 ns), za pristup NS polju,
- **uint8 GetPBit**(), čitanje polja P bita,
- **uint8 GetFBit**(), čitanje polja F bita,
- **void SetPFBit**(uint8 bitValue, FRAME_TYPE frame), postavljanje P/F bita.

4.2.5. Klase CtrlMessage i PSTNMessage

Za vreme razmene poruka između parnih procesa postoji mogućnost pojave grešaka kao posledica raznih poremećaja u celokupnom sistemu. Iako su u okviru drugog nivoa V5.1 sprege definisani mehanizmi za kontrolu pojave grešaka i ponovno slanje neispravnih okvira u V5.1 standardu je definisano rukovanje greškama u primljenim porukama.

Prva aktivnost po prijemu svake poruke od parnog procesa na trećem nivou je obrada poruke u skladu sa procedurom za proveru ispravnosti poruke. Ova procedura je precizno definisana u V5.1 standardu u kojem je opisano šta se proverava kao i redosled ovih provera koji je isti za AN i LE stranu V5.1 sprege.

Rezultat ove procedure je ili da se poruka ignoriše ili da se nastavi njena obrada. Ukoliko je rezultat provere da se poruka ignoriše ceo sistem se ponaša kao da poruka nikada nije ni primljena.

Kako na trećem nivou postoje dva protokola PSTN i CTRL, procedura provere poruke je u mnogim elementima slična ali postoje i razlike. Zato postoje i dve klase CtrlMessage i PSTNMessage koje izvršavaju proveru poruka onako kako je to definisano standardom.

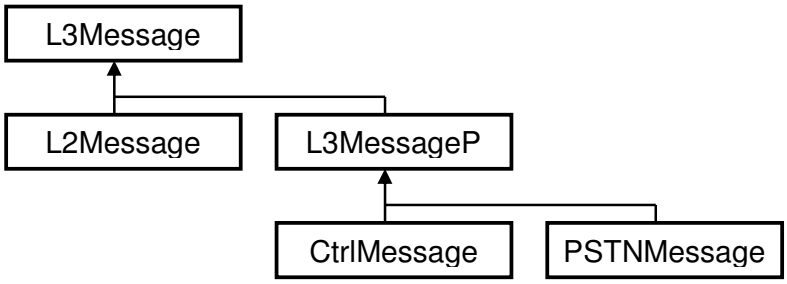
Zajednički elementi ovih klasa su implementirani u virtuelnoj klasi L3MessageP, koja sa druge strane nasleđuje klasu L3Message, dok su razlike implementirane u okviru funkcija koje obrađuju poruku.

Klasa PSTNMessage služi za obradu poruka koje stižu u PSTN protokol dok klasa CtrlMessage služi za obradu poruka CTRL protokola. Ove klase sadže funkcije koje samo utvrđuju prisustvo ili odststvo grešaka, kada se pojavi neka greška potrebno je formirati poruku o grešci koja se šalje upravljačkom podsistemu.

Poruke u sistemu automata mogu da formiraju i šalju samo objekti koji su automati zbog toga što su ove funkcije programskog jezgra dostupne samo ovim objektima. Iz ovog razloga klase PSTNMessage i CtrlMessage su spregnute sa odgovarajućim klasama na poseban način kako bi im bio omogućen pristup

programskom jezgru. Kako je programsko rešenje razvijeno u programskom jeziki C++ ova veza je učinjena upotrebom friend deklaracija.

Sve pomenute klase za obradu poruka u ovom programskom rešenju V5.1 sprege su međusobno povezane vezama koje su prikazane na slici 8.



Slika 8. Nasleđivanje klasa za obradu poruka u V5.1 sprezi

4.2.6. Ostale pomoćne klase

U ovu grupu spadaju sve ostale klase koje nemaju direktnu vezu sa programskim jezgrom ili sa V5.1 spregom. To su klase koje opisuju određene strukture podataka koje ulaze u sastav pojedinih klasa koje implementiraju automate V5.1 sprege.

Od ovih klasa najvažnije su:

- **Klasa DlBuffList**: ova klasa predstavlja specijalni red bafera, red je jako spregnut sa strukturom bafera koji se uzimaju po potrebi od programskog jezgra. Ovo je posledica toga što se koriste ista polja za smeštanje vrednosti pokazivača koja se koriste u programskom jezgru. Klasa ima standardne funkcije za dodavanje, pristup i brisanje elemenata iz liste i to su:
  - **bool Empty**(), testiranje da li je red prazan,
  - **void AddHead**(uint8 *ptrBuff), dodavanje bafera na početak strukture,
  - **void AddTail**(uint8 *ptrBuff), dodavanje bafera na kraj strukture,
  - **uint8 *GetHead**(), pristup prvom elementu ove strukture,
  - **uint8 *GetTail**(), pristup zadnjem elementu ove strukture,
  - **uint8 *RemoveHead**(), brisanje prvog elementa,
  - **uint8 *RemoveTail**(), brisanje zadnjeg elementa.

- **Klasa CircBuffer**: ovo je parametarska klasa što znači da se pri deklaraciji objekata ove klase mora definisati i klasa koja će predstavljati parametar. Ova klasa predstavlja cirkularni bafer pri čemu pomenuti parametar definiše tip podataka koji će biti smešten u ovaj bafer. Klasa se inicijalizuje sa jednim argumentom koji definiše maksimalan broj elemenata koji mogu da se smeste u ovaj bafer. U osnovi ima samo tri funkcije za rad i to su:
  - **bool Put**(T &ms), ovom funkcijom se dodaje novi element u cirkularni bafer koji je tipa T tj. novi element je instanca klase koja je definisana kao porametar prilikom deklaracije cirkularnog bafera,
  - **bool Get**(T &ms), ovom funkcijom je izbacuje prvi element koji je na redu iz cirkularnog bafera,
  - **bool IsEmpty**(), testira da li je cirkularni bafer prazan.

- **Klasa FifoBuffList**: predstavlja jednostavnu jednostruko spregnutu listu koja predstavlja jedan FIFO red (FIFO – First In First Out). Elementi ove liste su baferi koji se dobijaju od programskog jezgra. Ova klasa ima samo najosnovnije funkcije i to su:
  - **bool Empty**(), testiranje da li je lista prazna,
  - **void Add**(uint8 *ptrBuff), dodavanje bafera u listu,
  - **uint8 *Remove**(), izbacivanje elementa iz liste.

- **Klasa CDataStorePool**: je takođe parametarska klasa. Ova klasa predstavlja statičku strukturu koja ima funkciju da čuva objekte koji su instance klase koja se definiše parametrom kada se deklariše instanca ove klase. Po svojoj unutrašnjij strukturi se ponaša kao niz objekata date klase stim što je u ovu klasu ugrađena funkcionalnost praćenja referenci na objekte u ovoj strukturi podataka. Osnovne funkcije ove klase su:
  - **unsigned GetNumOfStores**(void), vraća broj objekata u smeštenih u ovoj strukturi podataka,
  - **T* GetDataStore**(unsigned storeId), vraća referencu objekta koji je u nizu smešten na poziciji koja je definisana argumentom storeId,
  - **T* GetReferencedDataStore**(unsigned reference), za referencu definisanu argumentom funkcija prvo pokušava da pronađe da li je u ovu strukturu podataka smešten objekat sa ovom referencom, ako nije funkcija vraća adresu prvog slobodnog mesta na koje dati objekat može da se smesti,
  - **unsigned ReferenceToStoreId**(unsigned reference), radi isto kao i predhodna funkcija stim da umesto adrese objekta vraća indeks u nizu objekata,
  - **unsigned StoreIdToReference**(unsigned storeId), vraća referencu na objekat koji je smešten u nizu na poziciji koja je definisana sa argumentom storeId,
  - **void SetReference**(unsigned storeId, unsigned reference), stavlja referencu objekta u nizu objekata na poziciju koja je definisana sa argumentom storeId,
  - **void ReleaseDataStore**(unsigned storeId), oslobađa mesto u nizu objekata koje se nalazi na poziciji koja je data argumentom storeId,
  - **void ReleaseReferencedDataStore**(unsigned reference), oslobađa mesto u nizu objekata na kom se nalazi objekat koji ima istu referencu kao i vrednost argumenta reference.

- **Klasa FileClass**: je pomoćna klasa koja se koristi samo prilikom testiranja. Osnovna funkcija ove klase je da omogući automatima pristup datoteci u koju se upisuju podaci o funkcionisanju pojedinih automata kao i celog sistema. Pristup definisanoj datoteci se odvija posredstvom kritične sekcije što omogućuje da automati beleže svoje podatke određenim redosledom. Najvaž niji atributi i funkcije ove klase su:
  - **CRITICAL_SECTION csFile**, kritična sekcija za kontrolu pristupa datoteci,
  - FILE *file, pokazivač na strukturu podataka koja predstavlja datoteku,
  - **char *fileName**, pokazivač na niz karaktera koji predstavljaju naziv datoteke,
  - **void Init**(char *newFileName), inicijalizacija objekta,
  - **void PutString**(char *string), upis u datoteku niza karaktera koji je dat argumentom string,
  - **void Open**(), otvaranje datoteke za upis karaktera,
  - **void Close**(), zatvaranje datoteke,

4.3. Fizički nivo V5.1 sprege
4.3.1. ISA KAN kartica

Za realizaciju fizičkog nivoa V5.1 sprege u oviru ovog programskog rešenja V5.1 sprege upotrebljena je ISA (ISA – Industry Standard Architecture) KAN kompjuterska kartica koja je razvijena na katedri za računarsku tehniku i računarske komunikacije.

Fizička arhitektura kartice zajedno sa odgovarajućom programskom podrškom (driver) realizuju sve mehaničke, električne, funkcionalne i proceduralne karakteristike 2048kbit/s (E1) linka, koji je opisan u ETS 300 166 i ETS 300 167 preporukama.

Da bi ISA KAN kartica se pravilno funkcionisala potrebno je izvršiti sledeće korake u navedenim redosledom:

- ugraditi ovu kartiicu u jedan od ISA konektora na matičnoj ploči,
- zatim je potrebno pokrenuti program za konfiguraciju BIOS-a (BIOS – Basic I/O System) gde je potrebno rezervisati IRQ 5 (IRQ – Interrupt request), IRQ 7 i IRQ 10 za ovu karticu kako bi programska podrška za ovu karticu pravilno funkcionisala,
- kao zadnji korak potrebno je ugraditi programsku podršku za ISA KAN karticu u sam operativni sistem.

Kada se svi ovi koraci izvrše pravilno ISA KAN kartica je spremna za rad. Pristup funkcijama ove kartice je omogućen kroz standardne funkcije operativnog sistema za rad sa perifernim uređajima.

U ovom programskom rešenju sprega između ISA KAN kartice i automata koji predstavljaju drugi nivo V5.1 sprege ostvarena je preko klase Driver**.** Programski kod za ovu klasu se nalazi u datoteci Ph_layer.cpp i zaglavljima Ph_layer.h i Ioctl.h.

4.3.2. Klasa Driver

U okviru ove klase definisani su potrebni atributi i funkcije koji zajedno omogućavaju inicijalizaciju komunikacije sa ISA KAN karticom kao i slanje i prijem nizova bajtova koji predstavlju poruke na višim nivoima.

Klasa Driver ima tri atributa, njihova nazivi i namena su:

- **unsigned short m_InBuffer[BUFFER_SIZE],** ovaj atribut predstavlja bafer za niz bajtova koji treba sa se pošalje parnom procesu kroz ISA KAN karticu, veličina bafera je ograničena vrednošću BUFFER_SIZE= 512 (1024 bajtova),
- **unsigned short m_OutBuffer[BUFFER_SIZE],** je bafer u koji se smešta niz bajtova koji je stigao sa parnog procesa u ISA KAN karticu,
- **HANDLE m_hndFile**, ovo je objekat preko koga se pristupa programskoj podršci ISA KAN kartice.

Osnovne funkcije članice klase Driver su:

- **bool OpenDriver**(), funkcija pokušava da upostavi komunikaciju sa programskom podrškom ISA KAN kartice, vraća vrednost true ako je akcija uspešna odnosno false ako nije uspostavljena komunikacija,
- **bool CloseDriver**(), funkcija prekida komunikaciju sa programskom podrškom ISA KAN kartice i vraća zauzete resurse operativnom sistemu ukoliko je pre ove funkcije uspešno bila pozvana funkcija OpenDriver(),

- **int driver**(int32 IoCtlCode, uint32 InBufferSize), ova funkcija se koristi za slanje komandi programskoj podršci ISA KAN kartice. Komande se šalju preko vrednosti argumenta dok se parametri za komande, ako oni postoje, prenose kroz atribut m_InBuffer. Ova funkcija radi u blokirajućem režimu što znači da se programski tok zaustavlja sve dokle god funkcija uspešno ili neuspešno ne završi svoju obradu. Definisane komade su:
  - IOCTL_Init, komanda za inicijalizaciju ISA KAN kartice,
  - IOCTL_GetVersion, komanda za proveru verzije,
  - IOCTL_SendData, komanda kojom se aktivira procedura za slanje podataka,
  - IOCTL_ReceiveData, komanda kojom se proverava da li ima podataka u baferu za prijem,
  - IOCTL_Debug , komanda za preuzimanje informacija o statusu fizičkog nivoa.
- **HANDLE GetDeviceViaInterface**(LPGUID lpGUID, int iDeviceInstance), ova funkcija se poziva samo prilikom inicijalizacije. Osnovni zadatak ove funkcije je da pristupi operativnom sistemu gde će pokušati da pronađe uređaj koji ima jedinstveni globalni identifikator isti kao vrednost argumenta lpGUID (za ISA KAN karticu ova vrednost je {3BF3581A-4CE1-44F3-A44E-98AE73C430DA}). Ako se funkcija uspešno izvrši povratna vrednost je objekat za rukovanje uređajem koji je bio tražen,
- **bool SendData**(uint16 Channel, uint8* buffer, uint16 MsgLen), ovom funkcijom se se šalju podaci kroz ISA KAN karticu,
- **bool ReceiveData**(uint16 Channel, uint8* buffer, uint16 &MsgLen), ovom funkcijom se primaju podaci koji su stigli u ISA KAN karticu.
- **bool PrintDebug**(char* msg), ovom funkcijom se preuzimaju poruke koje sadrže informacije o statusu fizičkog nivoa. Ova funkcija mora periodično da se poziva pošto se u toku rada dešavaju događaji koji generišu ove poruke pri čemu se svaka poruka smešta u poseban bafer. Kako se baferi uzimaju od programske podrške ISA KAN kartice ukoliko se isti nevrate, onog trenutka kada nema više slobodnih bafera dolazi do obaranja operativnog sistema i resetovanja računara.

Komunikacija drugog nivoa sa prvim nivo se ostvaruje tako što se formira jedan objekat koji je instnca klase Driver pri tome će se u okviru konstruktora klase pokrenuti inicijalizacija programske podrške ISA KAN kartice. Ako je ova inicijalizacija uspešno završena bez grešaka svi uslovi su ispunjeni za poziv funkcija za prijem i slanje podataka.

U programskom rešenju V5.1 sprege funkcija za slanje podataka se poziva od strane automata drugog nivoa po potrebi kada postoje podaci koje je potrebno poslati parnim procesima.

Prijem podataka od parnih procesa ostvaren je preko funkcija operativnog sistema za rukovanje vremenskim kontrolama. Ova procedura je realizovana tako što je definisana funkcija koja se periodično poziva od strane operativnog sistema a u okviru koje se nalazi poziv funkcije za prijem podataka. Ako postoje podaci u baferu za prijem podataka formira se odgovarajuća poruka koja se ubacuje u sistem automata i šalje se drugom nivou na obradu.

4.4. Nivo kanala podataka V5.1 sprege
4.4.1. LAPV5-EF podnivo

Osnovna funkcija podnivoa LAPV5-EF je prosleđivanje okvira između AN i LE strane V5.1 sprege. Ovo prosleđivanje podrazumeva spajanje signalizacionih puteva koji se prenose kroz isti komunikacioni kanal. Parni procesi razmenjuju poruke koje su smeštene u okvire čiji je format prikazan na slici 9.

8	7	6	5	4	3	2	1	Oktet
Početak okvira (FLAG)								1
EF adresa								2
EF adresa								3
Informacioni sadržaj								
Kontrolna suma (FCS)								N-3
Kontrolna suma (FCS)								N-2
Kraj okvira (FLAG)								N-1

Slika 9. Struktura okvira podnivoa za prenos paketa

Prikazana polja u okviru za prenos paketa imaju sledeće značenje:

- **Početak i kraj okvira** (FLAG), prvi oktet označava početak a zadnji oktet kraj okvira pri čemu zadnji oktet može da se iskoristi kao prvi oktet narednog okvira. Ovo polje uvek ima istu binarnu vrednost 01111110,
- **EF adresa** (Envelope function address), predstavlja adresno polje koje se sastoji od dva okteta koja su prikazana na slici 10.

8	7	6	5	4	3	2	1	Oktet
EF viša adresa						0	EA 1	2
EF niža adresa							EA 0	3

Slika 10. Format adresnog polja za prenos paketa

EF adresa se dobija spajanjem EF više adrese i EF niže adrese i čini adresu od 13 bita dok ostali biti imaju definisane vrednosti koje su uvek iste (EA – Extension Address). Vrednosti EF adrese imaju sledeće značenje:

- opseg vrednosti od 0 do 8175 koriste se za jedinstvenu identifikaciju ISDN korisničkih priključaka u okviru V5.1 sprege
- vrednosti od 8176 do 8191 su rezervisane i služe za identifikaciju pristupnih tačaka u kojima se obezbeđuju usluge nivoa kanala za mrežni nivo. Ova adresa je identična sa V5DL adresom u LAPV5-DL podnivou. U V5.1 protikolu koriste se sledeće vrednosti:
  - 8176 za PSTN signalizaciju,
  - 8177 za CTRL signalizaciju.

- **Informacioni sadržaj** (Information), predstavlja informacioni deo paketa odnosno poruke koje se razmenjuju između AN i LE strane V5.1 sprege. Informacioni deo može sadržati najmanje 3 okteta a naviše 533 okteta.
- **Kontrolna suma** (FCS – Frame Check Sequence), se sastoji od dva okteta i koristi se za detekciju grešaka.

4.4.2. Klasa LAPV5_EF

Realizacija podnivoa LAPV5-EF izvršena je kroz automat koji je implementiran u klasi LAPV5_EF. Ova klasa pored svoje osnovne funkcije izvršava i funkciju podnivoa LAPV5-FR kao i mapiranje između podnivoa LAPV5-EF i podnivoa LAPV5-FR i LAPV5-DL. Programski kod ove klase se nalazi u datoteci Lapv5ef.cpp i zaglavlju Lapv5ef.h.

Osnovni atributi ove klase koji su upotrebljeni u ovom programskom rešenju V5.1 sprege su:

- **L2Message m_L2Msg**, objekat klase L2Message koji se koristi za rukovanje porukama drugog nivoa,
- **uint8 m_NumOfPhLines**, ukupan broj fizičkih linija za signalizaciju,
- **uint8 PhLine**, broj fizičke linije kroz koju automat šalje signalizacione poruke,
- **uint32 m_idL2_CTRL**, adresa automata koji je tipa LAPV5_DL a koji je određen za prenos poruka CTRL protokola,
- **uint32 m_idL2_PSTN**, adresa automata koji je tipa LAPV5_DL a koji je određen za prenos poruka PSTN protokola,

Najvažnije funkcije članice ove klase su:

- **void InitFSM**(V51InterfaceNode *ptrV51, uint32 idL2_CTRL, uint32 idL2_PSTN), ova funkcija služi da inicijalizuje adrese objekata sa kojima ovaj automat komunicira. Funkcija je potrebna zbog zahteva da više instanci V5.1 sprege funkcionišu paralelno u istom sistemu autoamta.
- **void NORMAL_Main**(), ovo je jedina funkcija za obradu događaja pošto podnivo LAPV5-EF ima samo jednu funkciju a to je prosleđivanje okvira,
- **int GetValidEF**(uint8 phLineId, uint8* msg), funkcija se koristi za proveru EF adrese u primljenoj poruci. Ako EF adresa odgovara po konfiguraciji V5.1 sprege ovom automatu funkcija vraća vrednost EF adrese u suprotnom slučaju vraća se vrednost –1 što znači da se pojavila neka greška,
- **uint8 *AN_Frame_Relay**(uint16 ef, uint8* msg), ako je poruka poslata sa ISDN terminala ka LE strani V5.1 sprege funkcija dodaje EF adresu, definisanu konfiguracijom, u poruku a ako je poruka od LE strane ka ISDN terminalu funkcija izbacuje EF adresu iz poruke,
- **void PH_DATA_REQUEST** (uint16 len, uint8 *msg), funkcija se koristi za slanje paketa na prvi nivo.

4.4.3. LAPV5-DL podnivo

Osnovna funkcija podnivoa LAPV5-DL je da mrežnom nivou obezbedi uslugu pouzdane komunikacije uz proveru proveru ispravnosti poruka između AN i LE strane V5.1 sprege. Po strukturi ovaj podnivo odgovara LAPD protokolu koji je definisan u standardima ITU Q.920 i Q.921 kao i ETS 300 125.

Parni procesi ovog podnivoa međusobno razmenjuju informacije pomoću dve vrste okvira: okvir bez informacionog sadržaja (format A) i okvir sa informacionim sadržajem (format B).

Format ovih okvira prikazana je na slici 11.

8	7	6	5	4	3	2	1	Oktet
Link Adresa								1
Link Adresa								2
Control								3
Control								4

Format A

8	7	6	5	4	3	2	1	Oktet
Link Adresa								1
Link Adresa								2
Control								3
Control								4
Informacioni sadržaj								N

Format B

Slika 11. Format okvira podnivoa LAPV5-DL

1. Link Adresa, se sastoji od dva okteta pri čemu su polja unutar ove adrese definisana na sledeći način:

8	7	6	5	4	3	2	1	Oktet
V5DL viša adresa						C/R	EA 1	1
V5DL niža adresa							EA 0	2

Slika 12. Format adresnog polja Link Adress

- **V5DL** adresa je sastavljena od 13 bita i identična je vrednosti EF adrese u podnivou LAPV5-EF stim što se ne koriste vrednosti u opsegu od 0 do 8175 zato što se ove vrednosti koriste za identifikaciju ISDN korisnika. Adresa se koristi za identifikaciju protokaola na mrežnom nivou.
- **C/R** (Command/Response bit) je bit koji definiše da je okvir komandna ili odgovor. Ovaj bit se kodira na sledeći način:
  - Komanda:
    - okvir od LE prema AN strani C/R = 1,
    - okvir od AN prema LE strani C/R = 0,
  - Odgovor:
    - okvir od LE prema AN strani C/R = 0,
    - okvir od AN prema LE strani C/R = 1,
- **EA** predstavljaju bite za proširenje adrese i isti su za svaki okvir.

2. Control polje se sastoji od jednog okteta (U format) ili dva okteta (I format i S format) što zavisi od tipa poruke koja se šalje ovim okvirom. Navedena tri formata okvira su:
   - **I format** (Information Transfer Format) - se upotrebljava za prenosu poruka između parnih procesa mrežnog nivoa.
   - **S format** (Supervisory Format) - se koristi za prenos kontrolnih okvira koji se koriste za nadzor nivoa kanala u ove funkcije spadaju: potvrda prijema okvira I formata, zahtev za ponovno slanje I okvira i zahtev za privremeni prekid slanja I okvira.
   - **U format** (Unnumbered Format) - se koristi za obavljanje dodatnih kontrolnih funkcija nivoa kanala.

Na slici 13. prikazana je struktura okteta koji predstavljaju I, S i U format okvira.

Kotrolna polja	8	7	6	5	4	3	2	1	Oktet
I format	N(S)							0	4
	N(R)							P	5
S format	X	X	X	X	S	S	0	1	4
	N(R)							P/F	5
U format	M	M	M	P/F	M	M	1	1	4

Slika 13. Format kontrolnih polja

- **X**, predstavlja bite koji su rezervisani i uvek su postavljeni na vrednost 0,
- **M** i **S**, su biti koji definišu različite tipove okvira u S i U formatu,
- **P/F bit** (Poll/Final bit) definiše da li se za poslati okvir očekuje odgovor ili ne. Ovaj bit se u komandama naziva 'Poll' bit i postavlja se na vrednost 1 kada se zahteva od parnog procesa da vrati odgovor, a u odgovorima se naziva 'Final' bit i takođe se postavlja na vrednost 1.
- **N(S)** (Transmitter send sequence number) je broj I okvira koji se šalje parnom procesu.
- **N(R)** (Transmitter recevie sequence number) je broj I okvira koji se očekuje od parnog procesa.

Vrednosti polja N(S) i N(R) određuju se na osnovu vrednosti tri promenljive koje su pridružene LAPV5-DL podnivou na AN i LE strani, i to su: V(S), V(R) i V(A). Svaki I okvir se sekvencijalno broji, a vrednost brojača može biti u opsegu od: 0 do n-1 (gde je n moduo po kome se broje I okviri) za V5.1 spregu n = 127.

Vrednost V(S) (state variable) predstavlja broj sledećeg I okvira koji treba da se pošalje. Za svaki uspešno poslat okvir vrednost V(S) se povećava za 1, po modulu n. Vrednost V(S) ne sme preći vrednost V(A) za više od vrednosti k, pri čemu je k maksimalan broj nepotvrđenih prijema poslatih I okvira. Vrednost k se kreće u intervalu od: 1<=k<=(n-1). Vrednost N(S) se u trenutku slanja postavlja se na vrednost V(S).

Vrednost V(R) (receive state variable) predstavlja broj sledećeg I okvira koji se očekuje na prijemu. Povećava se za 1 nakon svakog uspešno primljenog okvira, po modulu n. Vrednost N(R) se u trenutku slanja I ili S okvira postavlja na vrednost V(R). Prijemnoj strani N(R) vrednost označava da su svi prethodni okviri do N(R)-1 ispravno primljeni i da se očekuje N(R) okvir.

Vrednost V(A) (acknowledge state variable) predstavlja broj poslednjeg potvrđenog I okvira od strane parnog procesa. Ova vrednost se ažurira nakon prijema svakog I ili S okvira sa ispravnom vrednošću N(R). Vrednost N(R) je ispravna ukoliko je je u opsegu: V(A)<=N(R)<=V(S).

Tipovi okvira u S formatu su:

- **RR** (Receive Ready), ova poruka se koristi za indikaciju spremnosti prijema okvira, za potvrdu prijema I okvira vrednostima N(S) manjim ili jednakim N(R)-1 i za uklanjanje uslova zauzetosti postavljenog ranijim slanjem RNR okvira od strane istog objekta nivoa kanala. RR komanda sa P bitom postavljenim na vrednost 1 može se koristiti za proveru stanja parnog procesa nivoa kanala.
- **REJ** (Reject) – ova poruka se koristi za indikaciju da je prethodno primljeni okvir odbačen te se zahteva ponovno slanje okvira počev od I okvira sa

brojem N(R). REJ komanda sa P bitom postavljenim na vrednost 1 može se koristiti za proveru stanja parnog procesa nivoa kanala.
- **RNR** (Receive Not Ready) – ovom porukom označava se trenutna nemogućnost prijema I okvira čiji je redni broj N(R). Za nastavak rada mora se poslati RR. RNR komanda sa P bitom postavljenim na vrednost 1 može se koristiti za proveru stanja parnog procesa nivoa kanala.

Tipovi okvira u U formatu su:
- **UA** (Unnumbered Acknowledgement) – poruka kojom se potvrđuje prijem i označava prihvatanje SABME komande. Ovaj odgovor ukida stanje zauzetosti postavljeno RNR okvirom,
- **DM** (Disconnect Mode) – poruka kojom se označava da nivo kanala nije u stanju da prihvati zahtev za uspostavu režima za razmenu više okvira (MFE – Multiple Frame Established),
- **SABME** (Set Asynchronous Balanced Mode Extended) – komanda kojom se zahteva uspostava režima za razmenu više okvira. Ako parni proces na ovu komandu odgovori sa UA komandom obe strane ulaze u MFE stanje. Samo u ovom stanju moguća je razmena informacionih okvira između parnih procesa.

3. Information, ovo polje predstavlja informacioni deo okvira podnivoa LAPV5-DL u okviru kojeg se vrši transport poruka sa mrežnog nivoa. Maksimalno može sadržati 260 okteta.

Poruke koje parni procesi ovog podnivoa razmenjuju su posledice primitiva koje se razmenjuju između ovog i njemu susednih nivoa, osnovne primitive su:
- Primitive koje se razmenjuju sa mrežnim nivoom:
  - DL_DATA_INDICATION, se koristi kada drugi nivo šalje poruke na treći nivo,
  - DL_DATA_REQUEST, se koristi kada treći nivo šalje poruke na drugi nivo,
- Primitive koje se razmenjuju se fizičkim nivoom:
  - PH_DATA_INDICATION, se koristi kada prvi nivo šalje poruke na drugi nivo,
  - PH_DATA_REQUEST, se koristi kada drugi nivo šalje poruke na prvi nivo,
- I primitive koje se razmenjuju sa upravljačkim podsistemom:
  - MDL_ESTABLISH_REQUEST, se koristi kada upravljački podsistem traži od drugog nivoa da započne proceduru za uspostavu režima za razmenu više okvira
  - MDL_ESTABLISH_INDICATION, se koristi kada upravljački podsistem javlja drugom nivou da je parni proces započeo proceduru za uspostavu MFE.
  - MDL_ESTABLISH_CONFIRAMTION, je primitiva za potvrdu ishoda procedure za uspostavu MFE,
  - MDL_RELEASE_INDICATION, je primitiva koja daje izveštaj o prekidu rada MFE procedure,
  - MDL_LAYER_1_FAILURE_INDICATION, predstavlja informaciju o stanju fizičkog nivoa,
  - MDL_ERROR_INDICATION, je primitiva kojom drugi nivo informiše upravljački podsistem da je došlo do greške u radu drugog nivoa. Upravljački podsistem je nadležan da dovede rad drugog nivoa u normalno stanje.

4.4.4. Klasa LAPV5_DL

Klasa LAPV5_DL predstavlja automat koji u potpunosti implementira podnivo LAPV5-DL. Osnovne funkcije koje sadrži ova klasa definisane su u standardu ETS 300 125 u okviru opisa SDL dijagrama kao i stanja koja ovaj automat treba da ima. Programski kod ove klase se nalazi u datoteci Lapv5dl.cpp i zaglavlju Lapv5dl.h.

Pored klase LAPV5_DL u zaglavlju Lapv5dl.h definisane su konstante i jedna pomoćna struktura koje su porebne za realizaciji automata LAPV5_DL. Navedene konstante su:

- **System_k_value** = 7, je maksimalan broj nepotvrđenih prijema poslatih I okvira
- **System_n_value** = 128, je moduo po kom se broje I okviri
- **SystemN200value** = 3, je maksimalan broj pokušaja da se ponovo pošalje okvir a za koji nije dobijena potvrda da je primljen od strane parnog procesa,
- **T200value** = 1, je vreme u sekundama koje automat čeka na odgovor od parnog procesa, ukoliko istekne ovo vreme ponovo se šalje okvir za koji nije dobijen odgovor od parnog procesa,
- **T203value** = 10, je maksimalno vreme u sekundama koje može da protekne a da nema razmene okvira između parnih procesa.

Struktura podataka sDlc (Data Link Connection) sadrži sve informacije koje su neophodne za komunikaciju parnih procesa nivoa kanala. Osnovni elementi ove strukture su:

- **uint8 PH_LineId**, je identifikator fizičkog kanala po kom se obavlja konekcija,
- **uint8 TxCbit, TxRbit, RxCbit, RxRbit**, su komandni (C) i bit odgovora (R) pri slanju i prijemu poruka,
- **uint16 ces** (Connection endpoint sufix), predstavlja vrednost preko koje upravljački podsistem adresira objekte nivoa kanala,
- **uint16 V5DLAddr**, sadrži vrednost V5DL adrese,
- **bool L3initiated**, indikacija da li je inicijalizovan odgovarajući objekat mrežnog nivoa,
- **bool dl_able_to_establish**, je promenljiva koja označava da li je objekat nivoa kanala u mogućnosti da uspostavi MFE režim sa parnim procesom;
- **uint8 VS, VA, VR**, su brojači koji je koriste za brojanje I okvira,
- **uint8 RC** (Retransmission count), je promenljiva koja je jednaka broju ponovljenih slanja okvira za koje nije dobijen odgovor od parnog procesa,
- **bool AcknowledgePending**, označava da je u toku potvrđivanje prijema okvira,
- **bool PeerReceiverBusy**, označava zauzetost parnog procesa na prijemu,
- **bool RejectException**, označava da li se desilo odbijanje okvira,
- **bool OwnReceiverBusy**, označava zauzetost objekta nivoa kanala na prijemu,
- **DlBuffList DataOutQ**, je red u kojem se čuvaju baferi sa porukama koje je potrebno poslati parnom procesu,
- **uint8* LastTransmitted**, ova promenljiva sadrži adresu zadnjeg okvira koji je poslat parnom procesu, koristi se ukoliko je potrebno ponovo poslati isti okvir zbog neispravnog prijema.

Stanja, atributi i najvažnije funkcije klase LAPV5_DL kao i opis ovih elemenata su prikazani u nastavku:

- Stanja automata LAPV5_DL su:
  - LINK_NOT_ESTABLISHED, je stanje u kojem nije uspostavljen kanal između parnih procesa dva podnivoa LAPV5-DL nivoa kanala.

- AWAITING_ESTABLISHMENT, je stanje u koje označava da je u toku procedura za uspostavu režima za razmenu više okvira,
- MULTIPLE_FRAME_ESTABLISHED, je stanje koje označava da su parni procesi uspostavili režim za razmenu više okvira i da je moguća razmena informacionih okvira,
- TIMER_RECOVERY, je stanje za oporavak vremenskih kontrola. U ovom stanju automat LAPV5_DL očekuje odziv na okvire koje se šalju parnom procesu.

- Osnovni atributi automata LAPV5_DL su:
  - Adrese automata u sistemu automata sa kojima automat LAPV5_DL razmenjuje poruke u okviru V5.1 sprege:
    - **uint32 m_idCC**, je adresa automata koji izvršava funkciju CTRL protokola,
    - **uint32 m_idLAPV5_EF**, je adresa automata podnivoa LAPV5-EF,
    - **uint32 m_idSYS**, je adresa automata koji u sastavu upravljačkog podsistema vrši funkciju upravljanja V5.1 spregom,
    - **uint32 m_idSYSL2PSTN**, je adresa automata koji u sastavu upravljačkog podsistema vrši upravljanje PSTN protokolom,
    - **uint32 m_idMNG**, je adresa automata koji u sastavu upravljačkog podsistema vrši funkciju obrade grešaka,
  - **sDlc dlc**, objekat sa podacima za komunikaciju parnih procesa nivoa kanala,
  - **L2Message m_L2InMsg**, je objekat koji se koristi za obradu poruka na drugom nivou a koje stižu od podnivoa LAPV5-EF,
  - **L2Message m_L2OutMsg**, je objekat koji se koristi za obradu poruka na drugom nivou a koje stižu sa mrežnog nivoa.

- Najvažnije funkcije automata LAPV5_DL su:
  - **void InitFSM**(uint16 ces, V51InterfaceNode *ptrV51, uint32 idCC, uint32 idMNG, uint32 idSYS, uint32 idLAPV5_EF, uint32 idSYSL2PSTN), funkcija služi za inicijalizaciju automata adresama objekata sa kojima će ovaj automat komunicirati u toku svog rada,
  - **void CleanUpFSM**(), funkcija vraća resurse koje je automat uzeo od programskog jezgra. Poziva se pre izbacivanja automata iz sistema automata,
  - **virtual void Process**(uint8* pMsg), ovo je redefinisana funkcija koja prvo proverava da li je primljena poruka ispravnog formata, a ako jeste tek se onda vrši obrada primljene poruke,
  - **void SendInternal**(uint16 event), ovom funkcijom automat sam sebi šalje poruke koje su definisane u standardu ETS 300 125,
  - **void SendToSYS**(uint16 mdl, uint16 err = 0), ovo je funkcija za slanje poruka upravljačkom podsistemu,
  - **void SendDlDataIndication**(), je funkcija za slanje poruka na mrežni nivo,
  - **bool FrameOK**(uint8* msg), je funkcija kojom se proverava ispravnost okvira koji su stigli od parnog procesa,
  - **uint8* FormatL3Msg**(uint8* msg), ovom funkcijom se poruke mrežnog nivoa ugrađuju u informacioni deo I okvira koje razmenjuju parni procesi podnivoa LAPV5-DL nivoa kanala,
  - **void Discardl_Queue**(), funkcija prazni izlazni red u koji se smeštaju I okviri koji su spremni za slanje parnom procesu,
  - **void EstablishDataLink**(), funkcija se koristi za uspostavu režima za razmenu više okvira slanjem SABME okvira parnom procesu,

- **void InvokeRetransmission**(uint8 NR), funkcija se koristi da izvrši ponovno slanje I okvira koji je zadnji bio poslat,
- **void ClearExceptionCondition**(), ovom funkcijom se ukidaju svi uslovi izuzeća,
- **void TransmitEnquiry**(), funkcija u slučaju da je objekat nivoa kanala zauzet prijemom salje RNR okvir komandu, ako nije RR okvir komandu,
- **void EnquiryResponse**(), funkcija u slučaju da je objekat nivoa kanala zauzet prijemom salje RNR okvir kao odgovor, ako nije RR okvir kao odgovor,
- **void PurgeRetranQ**(uint8 NR), funkcija izbacuje iz reda za slanje sve okvire ciji je prijem potvrdjen,
- **void NR_ErrorRecovery**(), je funkcija za oporavak pri pojavi greske NR polja pristiglog okvira,
- **int ModuloCompare**(uint8 A, uint8 B), je funkcija za komparaciju A i B vrednosti koje se racunaju po modulu n sa veličinom 'pozora' k,
- **bool NR_OK**(uint8 NR, uint8 VA, uint8 VS), funkcija se koristi za proveru da li je zadovoljen uslov VA<=NR<=VS,
- **void TX_SABME**(uint8 Pbit), ovom funkcijom se šalje SABME okvir,
- **void TX_UA**(uint8 Fbit), ovom funkcijom se šalje UA okvir,
- **void TX_DM**(uint8 Fbit), ovom funkcijom se šalje DM okvir,
- **void TX_RNR_COMMAND**(uint8 Pbit), funkcija šalje RNR komandu,
- **void TX_RNR_RESPONSE**(uint8 Fbit), funkcija šalje RNR odgovor,
- **void TX_RR_COMMAND**(uint8 Pbit), ovom funkcijom se šalje RR komanda,
- **void TX_RR_RESPONSE**(uint8 Fbit), ovom funkcijom se šalje RR odgovor,
- **void TX_I_COMMAND**(uint8 Pbit, uint8* msg), funkcija šalje I komandu,
- **void TX_REJ_RESPONSE**(uint8 Fbit), funkcija šalje REJ odgovor,
- **void Send_U_frame**(uint8 PH_LineId, uint8* msg), funkcija šalje U okvir automatu koji vrši funkciju podnivoa LAPV5_EF,
- **void Send_S_frame**(uint8 PH_LineId, uint8* msg), funkcija šalje S okvir automatu koji vrši funkciju podnivoa LAPV5_EF,

4.4.5. LAPV5-FR podnivo

Prenos signalizacionih poruka korisnika koji su priključeni preko ISDN korisničkih priključaka, sa AN strane V5.1 sprege, sprovodi se kroz podnivo LAPV5-FR koji izvršava funkciju prosleđivanja ISDN okvira kroz V5.1 spregu.

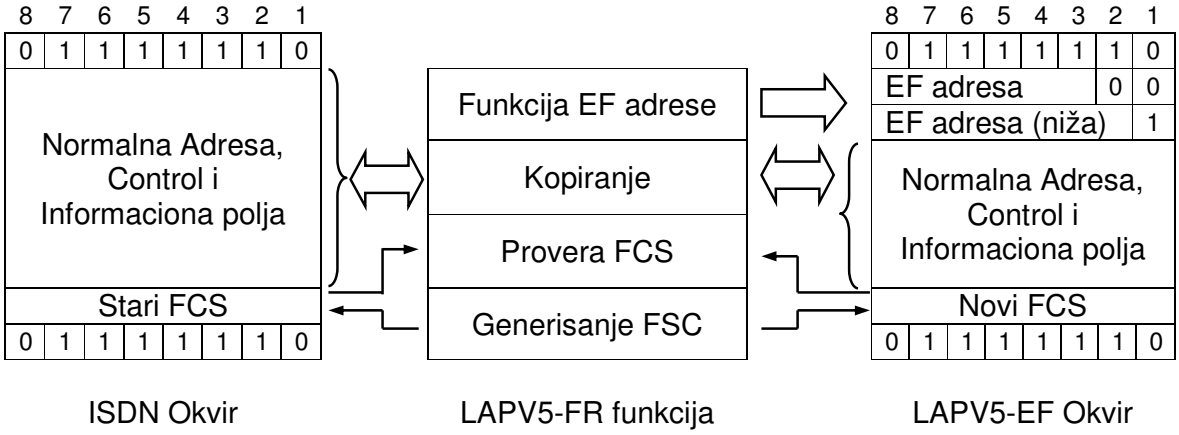
Ova funkcija sadrži nekoliko procedura:

- funkcije za ograničavanje, poravnavanje i transparentnost okvira,
- multipleksiranje i demultipleksiranje korišćenjem adresnog polja,
- proveru ispravnosti okvira.

Ispravan okvir, koji dođe sa nekog ISDN korisničkog priključka do AN, se prosleđuje kroz određeni vremenski odsečak na bazi ISDN adrese objekta nivoa kanala, nakon dodavanja EF adrese. Obrnuto, ispravan okvir koji dolazi sa LE se prosleđuje određenom ISDN korisničkom priključku nakon što se ukolni EF adresa.

Funkcije podnivoa LAPV5-FR u okviru ovog programskog rešenja V5.1 sprege su realizovane u okviru automata LAPV5-EF.

Na slici 14. je prikazana procedura za prosleđivanje okvira.



Slika 14. Funkcija podnivoa LAPV5-FR sa AN strane V5.1 sprege

4.4.6. Komunikacija između podnivoa

Komunikacija između podnivoa protokola nivoa kanala podrazumeva razmenu poruka između ovih podnivoa. Pri čemu treba da se obezbedi:

- komunikacija od **LAPV5-EF** ka **LAPV5-DL**,
  - proveru vrednosti polja EF adrese primljenog okvira i,
  - ukoliko je vrednost EF adrese unutar opsega rezervisanog za protokole mrežnog nivoa potrebno je da se prosledi informacioni deo okvira ka LAPV5-DL podnivou;
- komunikacija od **LAPV5-DL** ka **LAPV5-EF**,
  - preslikavanje poruka sa LAPV5-DL podnivoa u informacioni deo LAPV5-EF okvira i postavljanje polja EF adrese na vrednost V5DL adrese;
- komunikacija od **LAPV5-FR** ka **LAPV5-EF**,
  - prosleđivanje okvira nakon obrade u LAPV5-FR podnivou ka LAPV5-EF podnivo. Polje EF adrese postavlja se na vrednost EF adrese dodeljene ISDN korisničkom priključku;
- komunikacija od **LAPV5-EF** ka **LAPV5-FR**,
  - proveru vrednosti polja EF adrese primljenog okvira, te ukoliko je ta vrednost unutar opsega rezervisanog za identifikaciju ISDN korisničkog priključka prosledi informaciono i EF adresu okvira u LAPV5-FR podnivo radi dodatne obrade i daljeg prosleđivanja ka ISDN korisničkom priključku.

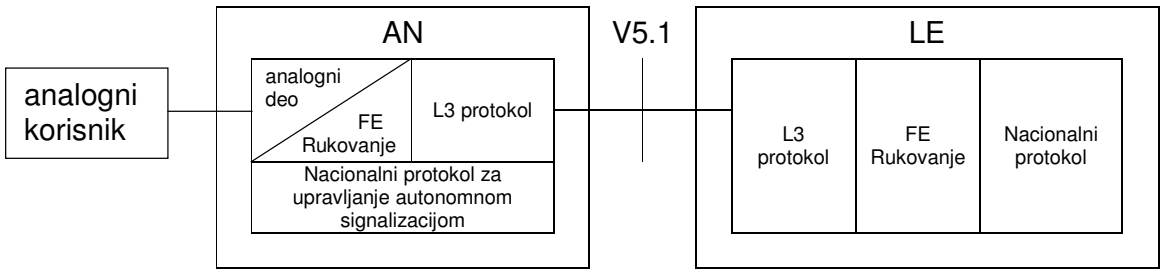
U ovom programskom rešenju V5.1 sprege sve navedene funkcije koje su potrebne za komunikaciju između podnivoa protokola nivoa kanala su realizovane u okviru automata LAPV5-EF.

4.5. Mrežni nivo V5.1 sprege
4.5.1. PSTN protokol

Osnovna funkcija PSTN protokola u okviru V5.1 sprege je da prenese signalizacione poruke od AN do LE čime se omogućava uspostavljanje i raskid poziva analognih korisnika.

PSTN protokol u osnovi predstavlja stimulativni protokol pošto se pozivima ne upravlja na AN strani, već se uz pomoć PSTN protokola na osnivu signalnih stimula generišu odgovarajuće poruke o stanjima linijskih signala i prosleđuju na LE stranu preko V5.1 sprege.

PSTN protokol u sastavu V5.1 sprege se koristi u sprezi sa nacionalnim protokolom. Nacionalni protokol na LE strani, koji se koristi za kontrolu korisničkih vodova direktno spojenih na LE, se takođe koristi i za kontrolu korisničkih vodova spojenih preko V5.1 sprege.



Slika 15. Funkcionalni model PSTN korisničkog priključka

PSTN ima mali broj funkcija koje obezbeđuju uspostavljanje signalnog puta, oslobađanje signalnog puta, i prenošenje signala stanja na analognom vodu.

Za vremenski kritične signalne sekvence, izdvojen je deo nacionalnog protokola sa LE strane na AN stranu odnosno određene signalne sekvence su ugrađene u AN deo PSTN protokola.

Većina signala na analognom vodu se ne inerpretirana od strane PSTN protokola nego se jednostavno prenosi od nacionalnog protokola sa AN strane do nacionalnog protokola na LE strani, i obrnuto, putem V5.1 sprege.

Stanja PSTN automata koji treba da implementira PSTN protokol mrežnog nivoa V5.1 sprege su sledeća:

- **Out Of Service (AN0)**, U ovo stanje automat će ući kada se pokrenu procedure ponovnog pokretanja (restart) od strane upravljačkog podsistema. Svi PSTN automati će ući u ovo stanje istovremeno.
- **Null State (AN1)**, Port je neaktivan, nema poziva koji je u toku.
- **Path Initiated By AN State (AN2)**, Detektovan je zahtev korisnika za uspostavu poziva, PSTN automat sa AN strane će poslati ESTABLISH poruku ka LE. AN sada čeka na ESTABLISH ACK poruku od LE strane.
- **Path Abort Request Sate (AN3)**, Poslata je ESTABLISH poruka od strane AN ka LE, i pre nego što je stigla poruka ESTABLISH ACK korisnik je prekinuo vezu.
- **Line Information State (AN4)**, U ovom stanju se automat nalazi dok LE obrađuje informacije o korisničkom priljučku. U ovo stanje se može preći samo iz Null State (AN1) stanja.
- **Path Active State (AN5)**, U ovom stanju automat razmenjuje signalne poruke.

- **Port Blocked State (AN6)**, U ovo stanje automat može preći iz bilo kog stanja. Kada port jednom pređe u ovo stanje jedino stanje u koje može preći je NULL stanje.
- **Disconnect Request State (AN7)**, AN traži od LE raskid veze. Ovo stanje se napušta kada se dobije potvrda raskida od LE strane ili ako isteknu vremenske kontrole, kada se informiše upravljanjački podsistem.

Format poruka koji se koristi za komunikaciju PSTN protokola mrežnog nivoa je prikazan na slici 16.

8	7	6	5	4	3	2	1	Oktet
Identifikator protokola								1
Adresa 3. nivoa (viša)							1	2
Adresa 3. nivoa (niža)								3
0	Tip poruke							4
Ostali informacioni elementi								itd.

Slika 16. Format PSTN poruka mrežnog nivoa V5.1 sprege

- **Identifikator protokola** (Protocol discriminator), ukazuje da se radi o poruci koja pripada nekom od mrežnih protokola V5.1 sprege. Vrednost ovog polja je uvek 01001000 binarno.
- **Adresa 3. Nivoa** (Layer 3 address), jednoznačno identifikuje PSTN korisnički priključak. Vrednost ovog polja sastoji se od 15 bita i može biti u osegu od 0 do 32767.
- **Tip poruke** (Message type), identifikuje poruku u PSTN protokolu. Mogući su sledeći tipovi poruka:
  - ESTABLISH – poruka kojom se zahteva uspostava veze,
  - ESTABLISH ACK – poruka kojom se odgovara na prethodno primljeni zahtev za uspostavu veze,
  - SIGNAL – poruka kojom se u određenom trenutku prenosi stanje neke linije na LE stranu, ili pak LE strana traži postavljanje određenog stanja neke linije na AN stranu,
  - SIGNAL ACK– poruka kojom se odgovara na prethodno primljenu SIGNAL poruku,
  - STATUS – poruka kojom se ukazuje na status PSTN protokola na AN strani. Ova poruka se šalje bilo na zahtev od strane LE strane (slanje STATUS_ENQUIRY) bilo na prijem neke neočekivane poruke od LE strane,
  - STATUS ENQUIRY – poruka koju LE strana šalje kada želi da zna kakav je status PSTN protokola na AN strani,
  - DISCONNECT – poruka kojom se zahteva raskid veze,
  - DISCONNECT COMPLETE – poruka kojom se odgovara na prethodno primljeni zahtev za raskid veze,
  - PROTOCOL PARAMETER – ovo poruku šalje LE strana kada želi da promeni parametre PSTN protokola na AN strani.
- **Ostali informacioni elemeti** (Other Information Elements), Stanja linije se opisuju nizom informacionih elemenata definisanih u PSTN protokolu u okviru V5.1 sprege, koji su detaljno opisani u ETS 300 324-1.

4.5.2. Klasa PSTNan

PSTN protokol mrežnog nivoa V5.1 sprege u okviru ovog programskog rešenja realizovan je u okviru klase PSTNan. Klasa PSTNan se koristi tako što se za svaki korisnički priključak instancira jedan automat.

Jedan od atributa klase PSTNan je objekat koji predstavlja instancu klase UserPortDef. Ovaj objekat sadrži podatke koji su specifični za stanje datog korisničkog priljučka kao i funkcije za realizaciju autonomnih signalnih sekvenci.

Osnovni elementi klase UserPortDef su:

- Najvažniji atributi ove klase:
  - **uint8 Ss**, brojač poslatih informacionih poruka,
  - **uint8 Sr**, brojač primljenih informacionih poruka,
  - **uint8 Sa**, brojač potvrđenih informacionih poruka,
  - **uint8 *ptrSavedFeMsg**, zadnja poruka polata parnom procesu koja je sačuvana da bi mogla kasnije ponovo da se pošalje u slučaju da istekne vremenska kontrola a nestigne odgovor od parnog procesa,
  - **uint8 response[5]**, pomoćni atribut koji se koristi za formatiranje informacionih elemenata koji kasnije ulaze u sastav poruka koje razmenjuje PSTN protokol,
  - **sAutoAck* AutoAck**, je struktura podataka u koju je smeštena trenutna konfiguracija autonomnog odziva PSTN automata na AN strani.
- Osnovne funkcije ove klase su:
  - **void RestSignalParams**(), funkcija resetuje vrednosti brojača Ss, Sr i Sa,
  - **uint8 *ClearPort**(), funkcija postavlja autonomni odziv na predefinisanu vrednost i odbacuje zadnju sačuvanu poruku ako je postojala,
  - **uint8 *GetSavedFeMsg**(), funkcija vraća zadnju sačuvanu poruku koju je odgovarajući PSTN automat poslao svom parnom procesu.
  - **uint8 *SaveFeMsg**(uint8 *msg), funkcija čuva poruku koja je zadnja poslata,
  - **uint8 *GetAutoAck**(uint8 param), funkcija vraća informacioni element koji se dobija na osnovu vrednosti argumenta param i strukture koja čuva autonomni odziv PSTN automata sa AN strane,
  - **void EnableAutoAck**(uint8 *param), funkcija omogućava autonomni odziv,
  - **void DisableAutoAck**(uint8 *param), funkcija se koristi kada je potrbno da se onemogući autonomni odziv,
  - **void SetAutoAckToDefault**(), funkcija resetuje autonomni odziv AN strane na unapred definisanu vrednost.

Pored navedene klase u sastav klase PSTNan su uključeni sledeći elementi:

- Vremenske kontrole:
  - **T1 (4s)**, ova vremenska kontrola se aktivira da bi se ograničilo vreme čekanja na poruke ESTABLISH ACK ili DISCONNECT COMPLETE kada se pošalje poruka ESTABLISH,
  - **T2 (20s),** ova vremenska kontrola se aktivira po isteku vremenske kontrole T1 a da nije dobijen odziv od parnog procesa, koristi se za kontrolu ponovnog slanja zahteva za uspostavu veze ili njenog prekida,
  - **T3 (2s),** vremenska kontrola T3 se aktivira kada se pošalje poruka DISCONNECT i predstavlja vreme čekanja na poruku DISCONNECT ili DISCONNECT COMPLETE ako se posle trećeg restartovanja ove vremenske kontrole ne dobije odgovor formira se poruka o grešci i šalje upravljačkom podsistemu,

  - **T4 (2s)**, Koristi se samo kada V5.1 sprega radi kao LE strana i predstavlja vreme čekanja posle slanja poruke STATUS ENQUIRY na poruku STATUS, posle trećeg isteka ove vremenske kontrole greška se prijavljuje upravljačkom podsistemu.
  - **Tr (5s)**, po prijemu poruke SIGNAL ili PROTOCOL PARAMETER pokreće se ova vremenska kontrola kojom se reguliše vreme nakon kojeg će biti poslat odgovor SIGNAL ACK,
  - **Tt (10s)**, nakon što se pošalje poruka SIGNAL ili PROTOCOL PARAMETER pokreće se ova vremenska kontrola kojom se reguliše vreme čekanja na odgovor SIGNAL ACK a po čijem isteku se šalje poruka DISCONNECT.
- Atributi u kojima se čuvaju vrednosti o stanju PSTN automata ili objekti koji se koriste za obradu poruka:
  - **UserPortDef UserPort**, objekat u kojem se čuvaju podaci o statusu automata,
  - **L3Message newMsg**, objekat za obradu poruka mrežnog nivoa,
  - **PSTNMessage pstnMsg**, objekat za rukovanje grešakama u porukama koje se razmenjuju između parnih procesa PSTN protokola,
  - **uint8 timesExpiredT3**, broj ponovnog startovanja vremenske kontrole T3,
  - **uint16 m_portId**, je broj PSTN korisničkog priključka koji je po konfiguraciji pridružen ovom PSTN automatu,
  - **uint32 m_idMNG**, je adresa automata koji u sastavu upravljačkog podsistema vrši funkciju obrade grešaka,
  - **uint32 m_idL2**, adresa automata koji je tipa LAPV5_DL a koji je određen za prenos poruka PSTN protokola,
  - **uint32 m_idRESTART**, je adresa automata u sastavu upravljačkog podsistema koji je zadužen za restartovanje V5.1 sprege,
  - **uint32 m_idCCAN**, je adresa automata koji vrši konverziju signalizacije nacionalnog protokola i signalizacije u V5.1 sprezi na AN strani.
- Finkcije PSTN automata:
  - **void InitFSM**(V51InterfaceNode *ptrV51, uint32 idMNG, uint32 idL2, uint32 idRESTART, uint32 idCCAN, uint16 portId), ova funkcija se koristi za inicijalizaciju PSTN automata kao bi se pravilno povezao za ostalim automatima koji pripadaju istoj V5.1 sprezi u okviru sistema automata,
  - **void CleanUpFSM**(), funkcija vraća preuzete resurse od programskog jezgra, koristi se prilikom izbacivanja automata iz sistema automata,
  - **bool IsOriginatingCallPrevail**(), ovom funkcijom se utvrđuje da li primljeni ili poslati poziv ima prioritet ukoliko se oni dogode u istom trenutku,
  - **void Process**(uint8* msg), ovo je redefinisana funkcija koja je u osnovnom obliku nasleđena iz programskog jezgra. Uvedena je zbog rukovanja greškama u porukama u skladu sa standardom za PSTN protokol u okviru ETS 300 324-1.
  - **void StopAllTimers**(), funkcija zaustavlja sve pokrenute vremenske kontrole,
  - Funkcije prelaza ovog automata u potpunosti preslikavaju procedure koje su date u SDL dijagramima koji se nalaze u standardu za V5.1 spregu ETS 300 324-1.

4.5.3. Klasa PupAn

Indikacija o statusu korisničkog priključka se zasniva na definisanoj podeli funkcija između AN i LE strane. Na AN strani se definišu funkcije za testiranje i održavanje korisničkog priključka dok su na LE strani definisane funkcije koje su važne za rukovanje korisničkim pozivima.

Upravljanje statusom korisničkih priključaka izvršava se pomoću automata koji su instance klase PupAn. Svakom korisničkom priključku pridružuje se jeadan PupAn automat koji preko svojih stanja definiše status korisničkog priključka.

Stanja PupAn automata se mogu podeliti u dve grupe neoperativna i operativna stanja što zavisi od vrste korisničkog priključka. Sva moguća stanja PupAn automata su:

- **Blocked (AN1.0),** korisnički priključak nije operativan,
- **Local Unblock (AN1.1),** ovo stanje označava da je AN strana pokrenula proceduru za prelazak u operativno stanje i čeka se odgovor od LE strane,
- **Remote Unblock (AN1.2),** ovo stanje označava da je LE strana pokrenula proceduru za prelazak u operativno stanje i čeka se odgovor od AN strane,
- **Operational Deactivated (AN2.0),** za PSTN korisnike ovo je operativno stanje dok za ISDN BA korisnike ovo stanje je operativno ali je digitalna sekcija deaktivirana,
- **Activation Initiated (AN2.1),** stanje u koje označava da je započeta aktivacija digitalne sekcije,
- **Access Activated (AN2.2)**, je stanje u koje je ISDN BA korisnički priključak operativan sa aktiviranom digitalnom sekcijom,
- **Pl Activation Initiated (AN3.1)**, je stanje za inicijalizaciju korisničkog priključka koji podržava komunikaciju preko iznajmljenih linija,
- **Pl Activated (AN3.2)**, je operativno stanje korisničkog priključka za iznajmljene linije,

Osnovni elementi automata PupAn su:

- Atributi
  - **uint32 m_idSYSPUP**, je adresa automata u sastavu upravljačkog podsistema koji je zadužen upravljanje korisničkim priključcima,
  - **uint32 m_idPC**, je adresa automata koji u sastavu CTRL protokola rukuje korisničkim priključcima,
  - **uint16 m_portId**, je redni broj korisničkog priključka kojem je ovaj automat pridružen,
  - **uint8 m_accessType**, tip korisničkog priključka,
- Funkcije članice:
  - **void InitFSM**(V51InterfaceNode *ptrV51, uint32 idSYSPUP, uint32 idPC, uint16 portId, uint8 accessType), funkcija služi za inicijalizaciju automata adresama i podacima koji su potrebni za pravilan rad ovog automata,
  - **void CleanUpFSM**(), funkcija vraća preuzete resurse od programskog jezgra,
  - **void SendToSYSPUP**(uint8 signalId), funkcija kojom se formiraju i šalju poruke automatu tipa SysPup koji se nalazi u sastavu upravljačkog podsistema,
  - **void SendToPC**(uint8 signalId), je funkcija kojom se formiraju i šalju poruke automatu tipa PC koji realizuje deo CTRL protokola,
  - Funkcije prelaza ovog automata su date u SDL dijagramima koji se nalaze u standardu za V5.1 spregu ETS 300 324-1.

4.5.4. CTRL protokol

CTRL protokol se koristi za koordinaciju upravljanja V5.1 sprege između AN i LE strane i sastoji se od dva dela koji izvršavaju dva tipa procedura:

- Upravljanje korisničkim priključcima PC (PC – Port Control). Dve najvažnije funkcije PC protokola su pojedinačno blokiranje i deblokiranje korisničkih priključaka,
- Zajedničko upravljanje CC (CC – Common Control) koje podrazumeva sledeće funkcije:
  - proveru i identifikaciju trenutne konfiguracije V5.1 sprege,
  - ponovno pokretanje PSTN protokola,
  - sinhronizaciju između AN i LE strane prilikom promene trenutne konfiguracije u novu konfiguraciju V5.1 sprege,

Realizacija CTRL protokola je izvedena upotrebom dva tipa automata PC i CC koji vrše gore pomenute funkcije. Automat CC je jedinstven za V5.1 spregu dok se PC automati povezuju jedan na jedan sa automatima koji definišu status individualnih korisničkih priključaka.

Stanja automata, koja su definisana u standardu V5.1 sprege, su identična za automate PC i CC, i to su:

- **Out of Service**, u ovo stanje se prelazi po prijemu komande MDU_stop_traffic od upravljačkog podsistema,
- **In Service**, predstavlja inicijalno i normalno stanje rada ovih automata,
- **Awaiting acknowledgement**, je privremeno stanje u koje ovi automati prelaze kada se pošalje neka poruka parnom procesu i traje do se nedobije potvrda o njenom prijemu,

Format poruke koje razmenjuju parni procesi između automata PC i CC u sastavu CTRL protokola mrežnog nivoa je prikazan na slici 17.

8	7	6	5	4	3	2	1	Oktet
Identifikator protokola								1
Adresa 3. nivoa (viša)								2
Adresa 3. nivoa (niža)								3
0	Tip poruke							4
Ostali informacioni elementi								itd.

Slika 17. Format poruka CTRL protokola mrežnog nivoa V5.1 sprege

- **Identifikator protokola** (Protocol discriminator), ukazuje da se radi o poruci koja pripada nekom od mrežnih protokola V5.1 sprege. Vrednost ovog polja je uvek 01001000 binarno.
- **Adresa 3. nivoa** (Layer 3 address), jednoznačno identifikuje poruke koje pripadaju CTRL protokolu mrežnog nivoa. Poruka koja pripada CC automatu sastoji se od 13 bita i uvek ima vrednost 8177. Poruka koja je namenjena za upravljanje statusom korisničkog priključka sastoji se od 13 bita ako je namenjena PC automatu koji upravlja sa ISDN BA korisničkim priključkom odnosno sastoji se od 15 bita ako je poruka namenjena PC automatu koji upravlja sa PSTN korisničkim priključkom. Format ovih polja prikazan je na slikama 18. i 19.

8	7	6	5	4	3	2	1	Oktet
Adresa 3. nivoa (viša)						0	0	1
Adresa 3. nivoa (niža)							1	2

Slika 18. Format adresnog polja poruke za CC automat i PC automat koji upravlja sa ISDN BA korisničkim priključkom

8	7	6	5	4	3	2	1	Oktet
Adresa 3. nivoa (viša)							1	1
Adresa 3. nivoa (niža)								2

Slika 19. Format adresnog polja poruke za PC autoamat koji upravlja PSTN korisničkim priključkom

- **Tip poruke** (Message type), identifikuje tip poruke u CTRL protokolu. Mogući su sledeći tipovi poruka:
  - PORT CONTROL, ovom porukom se razmenjuju upravljačke informacije za određeni PSTN ili ISDN BA korisnički priključak,
  - PORT CONTROL ACK, parni proces šalje ovu poruku kao potvrdu prijema poruke PORT CONTROL,
  - COMMON CONTROL, ovom porukom se razmenjuju upravljačke informacije koje nisu vezane za određeni korisnički priključak,
  - COMMON CONTROL ACK, parni proces šalje ovu poruku kao potvrdu prijema poruke COMMON CONTROL,
- **Ostali informacioni elementi** (Other Information Elements), predstavlja niz informacionih elemenata koji je specifičan za svaku poruku. Oni su definisani u CTRL protokolu u okviru V5.1 sprege, koji je detaljno opisan u ETS 300 324-1.

4.5.5. Klasa CommonControl

Klasa CommonControl predstavlja realizaciju konačnog automata CC koji se nalazi u sastavu CTRL protokola mrežnog nivoa V5.1 sprege. U sastavu ove klase nalaze se sledeći elementi koji su potrebni za njeno funkcionisanje:

- vremenska kontrola **T02 (1s)**, se koristi tako što se posle svake poslate poruke koja je tipa COMMON CONTROL aktivira ova vremenska kontrola. Vreme koje je potrebno da istekne ova vemenska kontrola predstavlja maksimalno vreme u toku kog mora da stigne potvrda prijema COMMON CONTROL ACK. Ako ova vremenska kontrola istekne automat CC će pokušati ponovo da pošalje prvobitnu poruku. Ukoliko vremenska kontrola istekne nakon što je drugi put poslata prvobitna poruka formira se poruka o grešci koja se šalje upravljačkom podsistemu.
- Osnovni atributi ove klase:
  - **CtrlMessage ccmsg**, je objekat koji se koristi za kontrolu i rukovanje sa greškama u porukama koje stižu od parnog procesa,
  - **L3Message oldMsg**, je objekat koji se koristi za rukovanje porukama mreznog nivoa u okviru V5.1 sprege,
  - **sCcStore CcStore**, je struktura podataka koja u svom satavu sadrži podatke koji su potrebni za rad CC automata. To su broj isteka vremenske kotrole T02,

zadnja poslata poruka koja se koristi u slučaju ponovnog slanja i red u koji se smeštaju poruke koje treba da se pošalju parnom procesu.
- **uint32 m_idL2**, je adresa automata tipa LAPV5-DL koji služi za prenos signalizacije CTRL protokola,
- **uint32 m_idRE**, je adresa automata koji služi za sinhronizovanu promenu konfiguracije V5.1 sprege,
- **uint32 m_idSYS**, je adresa automata u sastavu upravljačkog podsistema koji upravlja radom V5.1 sprege,
- **uint32 m_idMNG**, je adresa automata u sastavu upravljačkog podsistema koji se koristi za obradu grešaka koje se događaju u toku rada V5.1 sprege,
- **uint32 m_idRESTART**, je adresa automata koji vrši funkciju ponovnog startovanja PSTN protokola,

- Najvažnije funkcije članice ove klase su:
  - **void InitFSM**(V51InterfaceNode *ptrV51, uint32 idL2, uint32 idRE, uint32 idSYS, uint32 idMNG, uint32 idRESTART), ova funkcija inicijalizuje automat CC adresama ostalih automata u sistemu automata sa kojima on komunicira a koji pripadaju istoj instanci V5.1 sprege,
  - **void CleanUpFSM**(), funkcija vraća preuzete resurse od programskog jezgra,
  - **void Process**(uint8* msg), je redefinisana funkcija programskog jezgra kako bi se izvršila provera grešaka u pristiglim porukama od parnog procesa. Procedura za proveru je definisana u CTRL protokolu koji se nalazi u okviru standarda V5.1 sprege ETS 300 324-1,
  - **void SendToL2**(uint8 signalId, uint8 *msg = NULL), ovom funkcijom se formira i šalje poruka definisana argumentom signalId automatu nivoa kanala koji je određen za prenos poruka CTRL protokola,
  - **void SendToMNG**(uint8 signalId), funkcija formira i šalje poruko automatu koji služi za rukovanje greškama u V5.1 sprezi,
  - **void SendToSYS**(), funkcija u zavisnosti od parametara u primljenoj poruci šalje odgovor odgovarajućem automatu upravljačkog podsistema,
  - **void SaveMduCtrl**(uint8* msg), funkcija ulančava poruku datu argumentom msg u red za slanje poruka parnom procesu,
  - **uint8 *GetSavedMduCtrl**(), funkcija uzima poruku koja je na redu za salnje iz reda za slanje poruka parnom procesu,
  - **void ClearSavedMduCtrls**(), funkcija briše sve poruke koje su sačuvane u redu za slanje poruka parnom procesu,
  - **void SaveSentMsg**(uint8 *msg), funkcija čuva zadnju poslatu poruku tipa COMMON CONTROL dok ne stigne odgovor od parnog procesa da je primljena ili se poruka ne obriše,
  - **uint8 *GetSentMsg**(), funkcija vraća zadnju sačuvanu poruku tipa COMMON CONTROL, koristi se u slučaju da je potrebno ponovo poslati ovu poruku,
  - **void ClearSentMsg**(), funkcija briše zadnju poruku koja je bila čuvana,
  - **void IncT**(), funkcija uvećava za jedan vrednost promenljive u kojoj se čuva broj koliko puta je istekla vremenska kontrola T02,
  - **void ResetT**(), postavlja vrednost promenljive u kojoj se čuva broj koliko je puta istekla vremenska kontrola T02 na vrednost nula,
  - **bool FirstExpiry**(), funkcija vraća vrednost tačno ako je vremenska kontrola T02 istekla samo jedanput,
  - funkcije prelaza su implementirane u skladu sa SDL dijagramima koji su dati u standardu V5.1 sprege ETS 300 324-1.

4.5.6. Klasa PC

Klasa PC predstavlja realizaciju konačnog automata PC koji se nalazi u sastavu CTRL protokola mrežnog nivoa V5.1 sprege. U sastavu ove klase nalaze se sledeći elementi koji su potrebni za njeno funkcionisanje:

- vremenska kontrola **T01 (1s)**, ima identifnu funkcijau kao vremenska kontrola T02 u automatu CC. Jedina razlika je u tome što se vremenska kontrola T01 koristi sa porukom tipa PORT CONTROL a čeka se potvrda prijema sa porukom PORT CONTROL ACK.
- Osnovni atributi ove klase:
  - **CtrlMessage pcmsg**, je objekat koji se koristi za kontrolu i rukovanje sa greškama u porukama koje stižu od parnog procesa,
  - **L3Message oldMsg**, je objekat koji se koristi za rukovanje porukama mreznog nivoa u okviru V5.1 sprege,
  - **sPcStore PcStore**, je struktura podataka koja u svom satavu sadrži podatke koji su potrebni za rad PC automata. To su broj isteka vremenske kotrole T01, tip korisničkog pristupa, zadnja poslata poruka koja se koristi u slučaju ponovnog slanja i red u koji se smeštaju poruke koje treba da se pošalju parnom procesu.
  - **uint32 m_idL2**, je adresa automata tipa LAPV5-DL koji služi za prenos signalizacije CTRL protokola,
  - **uint32 m_idMNG**, je adresa automata u sastavu upravljačkog podsistema koji se koristi za obradu grešaka koje se događaju u toku rada V5.1 sprege,
  - **uint32 m_idPUP**, je adresa autoamta kojim se definiše stanje korisničkog priključka a koji je u direktnoj sprezi sa ovom instancom automata klase PC,
  - **uint16 m_portId**, je redni broj korisničkog priključka kojem je ovaj automat pridružen,
- Najvažnije funkcije članice ove klase su:
  - **void InitFSM**(V51InterfaceNode *ptrV51, uint32 idL2, uint32 idMNG, uint32 idPUP, uint16 portId, uint8 accessType), funkcija inicijalizuje PC automat,
  - **void CleanUpFSM**(),funkcija vraća preuzete resurse od programskog jezgra,
  - **void Process**(uint8* msg), funkcija programskog jezgra koja je redefinisana zbog obrade grešaka u porukama koje stižu od parnog procesa,
  - **void SendToL2**(uint8 signalId, uint8 *msg = NULL), funkcija formira i šalje poruku automatu LAPV5_DL koji prenosi poruke CTRL protokola,
  - **void SendToMNG**(uint8 signalId), funkcija kojom se formira i šalje poruka o grešci upravljačkom podsistemu,
  - **void SendToPUP**(), je funkcija kojom se šalje poruka PupAn automatu,
  - **uint8 GetFeCode**(uint8* msg), izdvaja funkcionalne elemente iz poruke,
  - **void SaveFeMsg**(uint8* msg), čuva poruku u redu za slanje parnom procesu
  - **uint8 *GetSavedFeMsg**(), uzima poruku iz reda za slanje parnom procesu,
  - **void ClearSavedFe**(), briše sve poruke u redu za slanje parnom procesu,
  - **void SavePortControl**(uint8* msg), čuva zadnju poslatu poruku,
  - **uint8 *GetPortControl**(), uzima zadnju poslatu poruku parnom procesu,
  - **void ClearSentMsg**(), briše zadnju poslatu poruku,
  - **void IncT**(), uvećava vrednost brojača koji broji koliko puta je istekao T01,
  - **void ResetT**(), postvalja vrednost pomenutog brojača na nulu,
  - **bool FirstExpiry**(); vraća vrednost tačno ako je T01 istekao samo jedanput,
  - funkcije prelaza su implementirane u skladu sa SDL dijagramima koji su dati u standardu V5.1 sprege ETS 300 324-1.

4.6. Upravljački podsistem V5.1 sprege

Osnovni zadatak upravljačkog podsistema je kontrola i sinhronizacija rada svih automata koji svojim radom realizuju funkcije V5.1 sprege. Upravljački podsistem je sastavljen od skupa automata pri čemu je svaki automat specijalizovan za upravljanje tačno određenim delom V5.1 sprege.

Automati upravljačkog podsistema zbog procedura i funkcija koje obavljaju nepripadaju nijednom od tri navedena nivoa V5.1 sprege već su povezani direktno sa svim automatima nad kojima izvršavaju funkciju upravljajnja u sistemu automata V5.1 sprege a preko njih sa odgovarajućim automatima koji izvršavaju iste funkcije na drugoj strani V5.1 sprege.

U okviru ovog programskog rešenja V5.1 sprege svi automati upravljačkog podsistema imaju po jednu instancu u okviru jedne V5.1 sprege. Automati upravljačkog podsistema su implementirani u sastavu odgovarajućih klasa i to su:

- **Klasa SysManagement**, implementira funkcije automata SysManagement. Programski kod ove klase se nalazi u datoteci An_Sys.cpp i zaglavlju An_Sys.h. Osnovna funkcija klase SysManagement je da izvršava sledeće procedure upravljanja:
  - postupak pravilnog startovanja V5.1 sprege, ovo podrazumeva inicijalizaciju odgovarajućih automata tačno definisanim redosledom uz primenu odgovarajućih vremenskih kontrola koje upravaljaju tokom ove procedure,
  - procedure za uspostavljanje i raskid komunikacionih puteva na nivou kanala za prenos podataka CTRL protokola mrežnog nivoa, ovo podrazumeva poziv odgovarajućih primitiva koje dovode nivo kanala u stanje režima za razmenu više poruka i zatim inicijalizaciju automata koji izvršava funkciju CTRL protokola koji je posle ove procedure doveden u stanje normalnog rada,
  - proveru konfiguracije V5.1 sprege preko funkcija CTRL protokola, prilikom svakog startovanja V5.1 sprege neophodna je provera koja će potvrditi da AN i LE strana rade sa istom konfiguracijom V5.1 sprege.

- **Klasa PstnDlManagement**, implementira funkcije automata PstnDlManagement. Programski kod ove klase se nalazi u datoteci Pstnmng.cpp i zaglavlju Pstnmng.h. Osnovna funkcija klase PstnDlManagement je da izvršava sledeće procedure upravljanja :
  - procedure za uspostavljanje i raskid komunikacionih puteva na nivou kanala za prenos podataka PSTN protokola mrežnog nivoa, ovo podrazumeva da će se ukoliko je konfiguracijom definisano da postoje PSTN korisnički priključci pokrenuti poziv odgovarajućih primitiva koje dovode nivo kanala u stanje režima za razmenu više poruka a da će se zatim inicijalizovati automati koji izvršavaju funkciju PSTN protokola koji će posle ove procedure biti dovedeni u stanje normalnog rada,
  - pravilna inicijalizacija PSTN protokola i svih odgovarajućih korisničkih priključaka,

- **Klasa Restart,** implementira funkcije automata Restart. Programski kod ove klase se nalazi u datoteci Restart.cpp i zaglavlju Restart.h. Osnovna funkcija klase Restart je da izvršava sledeće procedure upravljanja :
  - proceduru za startovanje i po potrebi ponovno startovanje svih PSTN protokola koji su definisani trenutnom konfiguracijom V5.1 sprege,

  - sinhronizuje proceduru startovanja PSTN protokola preko CTRL protokola na AN i LE strani V5.1 sprege,

- **Klasa ReAn,** implementira funkcije automata ReAn. Programski kod ove klase se nalazi u datoteci Re.cpp i zaglavlju Re.h. Osnovna funkcija klase ReAn je da izvršava sledeće procedure upravljanja :
  - verifikacija trenutne konfiguracije, ovo podrazumeva razmenu informacionih elemenata između AN i LE strane koji sadrže podatke o trenutnoj varijanti i jedinstvenom identifikacionom broju,
  - sadrži procedure koje preko stanja ovog automata definišu spremnost V5.1 sprege da promeni svoju trenutnu konfiguraciju,
  - sinhronizuje proces promene konfiguracije V5.1 sprege,

- **Klasa SysRe,** implementira funkcije automata SysRe. Programski kod ove klase se nalazi u datoteci SysRe.cpp i zaglavlju SysRe.h. Osnovna funkcija klase SysRe je da izvršava sledeće procedure upravljanja :
  - proceduru za promenu trenutne konfiguracije V5.1 sprege, za razliku od automata ReAn koji vodi računa o koordinaciji AN i LE strane automat SysRe služi za upravljanje celokupnom procedurom,

- **Klasa SysPup,** implementira funkcije automata SysPup. Programski kod ove klase se nalazi u datoteci SysPup.cpp i zaglavlju SysPup.h. Osnovna funkcija klase SysPup je da izvršava sledeće procedure upravljanja :
  - upravlja automatima koji predstavljaju status korisničkih priključaka,

- **Klasa Mng,** implementira funkcije automata Mng. Programski kod ove klase se nalazi u datoteci Mng.cpp i zaglavlju Mng.h. Osnovna funkcija klase Mng je da izvršava sledeće procedure upravljanja :
  - prijem i obrada poruka o greškama u okviru V5.1 sprege u toku njenog rada,
  - preko ovog automata sprovodi se slanje poruka koje predstavljaju komande i izvršavaju se funkcije koje služe za testiranje rada V5.1 sprege.

5. Testiranje programskog rešenja V5.1 sprege

5.1. Uvod

Testiranje ovog programskog rešenja V5.1 sprege sprovedeno je upotrebom dva programa V51DLL i V51NWKAN. Ova dva programa nastala su implementacijom procedura koje su definisane u standardima ETS 300 324-4 i ETS 300 324-8.

Standardi su proizašli iz potrebe da se precizno defiše skup testnih slučajeva dovoljnih za ispunjavanje funkcionalnih zahteva definisanih standardom V5.1 sprege ETS 300 324-1.

Program V51DLL je dizajniran da u potpunosti proveri funkcionalnost nivoa kanala V5.1 sprege odnosno svih njegovih podnivoa, ovo je ostvareno kroz 145 testova koji se nalaze u sastavu ovog programa.

Program V51NWKAN ima zadatak da proveri funkcionalnost mrežnog nivoa V5.1 sprege odnosno svih protokola koji su u njemu definisani, što je realizovano kroz 181 test koji su definisani u okviru ovog programa.

Testiranje je izvedeno upotrebom dva računara pri čemu su programi V51DLL i V51NWKAN simulirali ponašanje LE strane V5.1 sprege dok je programsko rešenje realizovano tokom izrade ovog rada predstavljalo AN stranu V5.1 sprege.

Krajnji korisnici koji su u normalnom radu priljučeni sa AN strane V5.1 sprege su izostavljeni ali je njihovo prisustvo simulirano u okviru programskog rešenja preko funkcija koje simuliraju odgovarajuće događaje koji stižu sa korisničkih priključaka.

Proces testiranja se sastoji od niza koraka koji se ponavljaju dokle god se ne izvrše svi testovi u datom programu. Ovaj postupak je sledeći:

1. Na prvom računaru se pokrene program za testiranje sa odgovarajućim argumentom kojim se definiše test kojim se testira određena funkcionalnost programskog rešenja V5.1 sprege,

2. Na drugom rašunaru se pokrene programsko rešenje V5.1 sprege čija se funkcionalnost testira pri čemu je konfiguracija V5.1 sprege definisana testom koji se izvršava. Za testove gde se testira funkcionalnost ISDN BA korisničkih priključaka postojaće samo jedan ISDN BA priključak isto tako i za PSTN korisničke priključke ukoliko postoje, V5.1 sprega biće konfigurisana samo sa jednim PSTN korisničkim priključkom,

3. Nakon što se oba programa učitaju u operativnu memoriju na prvom računaru se pokreće test a na drugom se izvršava procedura startovanja V5.1 sprege,

4. Program za testiranje prikazuje izveštaj o toku testiranja na ekranu, i to prikazuje preliminarne ocene o samom toku testiranja i po potrebi zahteva aktiviranje funkcija za testiranje koje su definisane u samom programskom rešenju. Na kraju testa se prikazuje konačna ocena. Ova ocena može biti da je test bio uspešan (PASS) ili da je test bio neuspešan (FAIL).

5. Ukoliko je testiranje bilo neuspešno za pronalaženje grešaka postoje datoteke u kojima je vođena evidencija o toku testiranja i stanjima pojedinih automata. Na osnovu informacija iz ovih datoteka moguće je ustanoviti tačno mesto pojave nepravilnog rada programskog rešenja.

6. Nakon što se greška, otkrivena testom, otkloni ova procedura se ponavlja doklegod ocena testa ne pokaže da je test bio uspešan.

Redosled testova je definisan standardom.Prvo se izvršavaju testovi LAPV-EF podnivoa, zatim LAPV5-DL podnivoa a na kraju testovi koji se odnose na module mrežnog nivoa. Svaki test ima jedinstvenu oznaku gde pojedini delovi imaju sledeće značenje:

- identifikator testa za V51DLL: TC <t1><c><tpt><s><ss><nn>
- identifikator testa za V51NWKAN: TC <t2><c><g1><g2><tpt><s><ss><nn>
  - <t1>, tip protokola: 1. LAPV5-EF, 2. LAPV5-DL
  - <t2>, tip protokola: 1. CTRL, 2. PSTN
  - <c>, kategorija:
    - 1. Osnovna komunikacija (IT),
    - 2. Osnovne funkcije (CA),
    - 3. Ispravan odziv (BV),
    - 4. Neočekivane poruke (BO),
    - 5. Rukovanje greškama (BI),
    - 6. Testiranje vremenskih kontrola (TI),
  - <g1>, grupa: 1. CC, 2. PC, _ označava da vrednost nije bitna,
  - <g2>, podgrupa:
    - 1. Prenos poruka,
    - 2. Procedure upravljačkog podsistema za ponovno pokretanje V5.1 sprege
    - 3. Procedure upravljačkog podsistema za promenu konfiguracije i proveru ispravnosti trenutne konfiguracije V5.1 sprege,
    - 4. Upravljanje sa PSTN korisničkim priključcima,
    - 5. Upravljanje sa ISDN BA korisničkim priključcima,
    - _ označava da podgrupa nije relevantna za dati test
  - <tpt>, tip testa: S. samostalni test
  - <s>, stanje automata: (0-9), M znači da test obuhvata više stanja, i _ označava da stanje automata nije bitno,
  - <ss>, pod stanje: (0-9), _ označava da podstanje automata nije bitno,
  - <nn>, redni broj testa: (00-99),

U okviru ovih programa definisane su specijalne funkcije koje su potrebne kako bi se testirali svi elementi programskog rešenja V5.1 sprege. Ove funkcije pripadaju nekoj od grupa:

- Funkcije za promenu stanja automata, ove funkcije sadrže skup poruka kojima se željeni automat prevodi iz jednog stanja u drugo,
- Procedure za inicijalizaciju testova (Preambles) ove procedure sadrže niz odgovarajućih funkcija za promenu stanja automata koji se testira i dovode ostale automate u programskom rešenju koje se testira u odgovarajuća stanja u kojima treba da se započne testiranje,
- Procedure za vraćanje svih automata u inicijalno stanje (Postambles), posle svakog testa automati u programskom rešenju se dovode u inicijalno stanje,
- Funkcije za proveru stanja automata programskog rešenja koje se testira, se koriste za potvrdu o ispravnoj promeni stanja automata koji se nalaze u sastavu V5.1 sprege,
- Standardne funkcije koje se ponavljaju u okviru više testova,

Testiranje paralelnog i nezavisnog rada više instanci V5.1 sprege izvršeno je upotrebom ova dva programa. Ovo testiranje je izvedeno tako što su startovane tri instance V5.1 sprege u okviru sistema automata programskog jezgra pri čemu je samo jedna instanca V5.1 sprege bila testirana sa pomenuta dva programa.

5.2. Testiranje sa programom V51DLL

Testiranje nivoa kanala V5.1 sprege definisano je u standardu ETS 300 324-8 a realizovano programom V51DLL. U sastavu ovog programa testovi su grupisani na sledeći način:

- **Testovi podnivoa LAPV5-EF** koji se koriste za proveru funkcionalnosti ovog podnivoa uključujući tu i podnivoe LAPV5-FR i podnivo za mapiranje putanja. Testovi u sastavu ove grupe se dalje dele u zavisnosti od funkcionalnosti koju testiraju:
  - **Basic Interconnection (IT)**, osnovna komunikacija AN i LE strane, ovim testovima se utvrđuje da su ispunjeni osnovni preduslovi za komunikaciju parnih procesa ovog podnivoa AN i LE strane i da odgovarajući konfiguracioni parametri imaju ispravne vrednosti,
  - **Valid Behaviour (BV)**, ispravan odziv, podrazumeva da se u toku ovih testova razmenjuju ispravne sekvence poruka pri čemu je sadržaj tih poruka ispravan,
  - **Invalid Behaviour (BI)**, rukovanje porukama koje sadrže greške, što znači da programsko rešenje V5.1 sprege detektuje i postupa sa greškama u skladu sa odgovarajućim definicijama u standardu za V5.1 spregu,

- **Testovi podnivoa LAPV5-DL** koji se koriste za proveru funkcionalnosti ovog podnivoa. Testovi u sastavu ove grupe se dalje dele u zavisnosti od funkcionalnosti koju testiraju:
  - **Basic Interconnection (IT)**, osnovna komunikacija AN i LE strane, ovim testovima se utvrđuje da su ispunjeni osnovni preduslovi za komunikaciju parnih procesa ovog podnivoa AN i LE strane i da odgovarajući konfiguracioni parametri imaju ispravne vrednosti,
  - **Capability (CA)**, testiranje osnovnih funkcija podnivoa LAPV5-DL nivoa kanala,
  - **Valid Behaviour (BV)**, ispravan odziv, podrazumeva da se u toku ovih testova razmenjuju ispravne sekvence poruka pri čemu je sadržaj tih poruka ispravan ovi testovi se sprovode za sva stanja automata LAPV5_DL,
  - **Inopportune Behaviour (BO),** neočekivane poruke, podrazumeva rukovanje porukama koje imaju ispravn sadržaj ali koje su pristigle u neočekivanom trenutku ovi testovi se sprovode za sva stanja automata LAPV5_DL,
  - **Invalid Behaviour (BI)**, rukovanje porukama koje sadrže greške, podrazumeva da programsko rešenje V5.1 sprege detektuje i postupi sa greškama u skladu sa odgovarajućim definicijama u standardu za V5.1 spregu, ovi testovi se sprovode za sva stanja automata LAPV5_DL,
  - **Timer (TI)**, testovi koji proveravaju ponašanje i funkcionisanje vremenskih kontrola koje nalaze u sastavu automata LAPV5_DL,

5.3. Testiranje sa programom V51NWKAN

Testiranje mrežnog nivoa V5.1 sprege definisano je u standardu ETS 300 324-4 a realizovano programom V51NWKAN. U sastavu ovog programa testovi su grupisani na sledeći način:

- **Testovi CTRL protokola** koji se koriste za proveru funkcionalnosti ovog protokola. Testovi u sastavu ove grupe se dalje dele u zavisnosti od funkcionalnosti koju testiraju:
  - **Basic Interconnection (IT)**, osnovna komunikacija AN i LE strane, ovim testovima se utvrđuje da su ispunjeni osnovni preduslovi za komunikaciju parnih procesa CTRL protokola AN i LE strane i da odgovarajući konfiguracioni parametri imaju ispravne vrednosti,
  - **Capability (CA)**, testiranje osnovnih funkcija CTRL protokola mrežnog nivoa,
  - **Valid Behaviour (BV)**, ispravan odziv, podrazumeva da se u toku ovih testova razmenjuju ispravne sekvence poruka pri čemu je sadržaj tih poruka ispravan, testovi ove grupe se dele na dve podgrupe: testove koji testiranju automat koji izvršava COMMON CONTROL funkcije i testiranje automata koji izvršava PORT CONTROL funkcije,
  - **Inopportune Behaviour (BO),** neočekivane poruke, podrazumeva rukovanje porukama koje imaju ispravn sadržaj ali koje su pristigle u neočekivanom trenutku ovi testovi se sprovode za oba automata CTRL protokola,
  - **Invalid Behaviour (BI)**, rukovanje porukama koje sadrže greške, što znači da programsko rešenje V5.1 sprege detektuje i postupa sa greškama u skladu sa odgovarajućim definicijama u standardu za V5.1 spregu, testovi se sprovode za oba automata CTRL protokola,
  - **Timer (TI)**, testovi koji proveravaju ponašanje i funkcionisanje vremenskih kontrola koje nalaze u sastavu automata CC i PC,

- **Testovi PSTN protokola** koji se koriste za proveru funkcionalnosti PSTN protokola mrežnog nivoa V5.1 sprege. Testovi u sastavu ove grupe se dalje dele u zavisnosti od funkcionalnosti koje testiraju:
  - **Basic Interconnection (IT)**, osnovna komunikacija AN i LE strane, ovim testovima se utvrđuje da su ispunjeni osnovni preduslovi za komunikaciju parnih procesa PSTN protokola AN i LE strane i da odgovarajući konfiguracioni parametri imaju ispravne vrednosti,
  - **Capability (CA)**, testiranje osnovnih funkcija PSTN protokola mrežnog nivoa V5.1 sprege,
  - **Valid Behaviour (BV)**, ispravan odziv, podrazumeva da se u toku ovih testova razmenjuju ispravne sekvence poruka pri čemu je sadržaj tih poruka ispravan ovi testovi se sprovode za sva stanja automata PSTNan,
  - **Inopportune Behaviour (BO),** neočekivane poruke, podrazumeva rukovanje porukama koje imaju ispravn sadržaj ali koje su pristigle u neočekivanom trenutku ovi testovi se sprovode za sva stanja automata PSTNan,
  - **Invalid Behaviour (BI)**, rukovanje porukama koje sadrže greške, podrazumeva da programsko rešenje V5.1 sprege detektuje i postupi sa greškama u skladu sa odgovarajućim definicijama u standardu za V5.1 spregu, ovi testovi se sprovode za sva stanja automata PSTNan,
  - **Timer (TI)**, testovi koji proveravaju ponašanje i funkcionisanje vremenskih kontrola koje nalaze u sastavu automata PSTNan,

6. Zaključak

U okviru ovog diplomskog rada prikazana je realizacija programskog rešenja koje obezbeđuje paralelan i međusobno nezavisan rad više instanci V5.1 sprege. Programsko rešenje je realizovano tako da se pojedine V5.1 sprege mogu posebno konfigurisati.

Programsko rešenje sadrži sve tražene funkcije. Pre svega tu su funkcije koje omogućuju instanciranje nove V5.1 sprege, povezivanje iste sa elementima fizičke arhitekture, pridruživanje raznih konfiguracija svakoj V5.1 sprezi posebno kao i funkcije za puštanje V5.1 sprege u rad, njeno zaustavljanje i brisanje iz celog sistema.

Klasa Mng pored svoje osnovne funkcije u okviru upravljačkog podsistema V5.1 sprege poseduje odgovarajuće funkcije članice preko kojih se šalju odgovarajuće poruke i simuliraju odgovarajući događaji koji potiču izvan V5.1 sprege. Ove funkcije su iskorišćene kako bi se potvrdila tražena funkcionalnost kroz testiranje, koju je ovo programsko rešenje trebalo da ispuni.

V5.1 sprega je implementirana u okviru ovog programskog rešenja u skladu sa standardima G.964 od strane ITU (koji se poziva na preporuke Q.920 i Q.921) odnosno ETS 300 324-1 od strane ETS (koji se poziva na preporuku ETS 300 125). Realizacija V5.1 sprege izvedena je primenom konačnih automata što je ostvareno preko progamskog jezgra za sistemsku podršku u realnom vremenu za konačne determinističke automate.

Programski kod koji je predstavljao polaznu osnovu za realizaciju V5.1 sprege a koji se oslanjao na staro programsko jezgro uspešno je transformisan u programski kod koji se u potpunosti oslanja na novo programsko jezgro pri čemu je zadržana funkcionalnost V5.1 sprege ali je ceo projekat proširen kako bi se zadovoljili funkonalni zahtevi koji su postavljeni u ovom zadatku.

Programsko rešenje V5.1 prikazano u ovom radu realizovano je da podržava rad sa više programskih niti što je potreban uslov za paralelan rad više instanci V5.1 sprege. Svi automati nalaze se u sistemu automata. Sistem automata predstavlja objekat klase FSMSystem koji upravlja resursima programskog jezgra i vodi računa da ceo sistem pravilno funkcioniše.

Elementarna komunikacija između korisnika i ovog programskog rešenja ostvarena je preko skupa funkcija koje predstavljaju komande i poruke koje se šalju odgovarajućim komponentama ovog programskog rešenja.

Testiranje prikazanog programskog rešenja kao i V5.1 sprege u njenom sastavu sprovedeno je u skladu sa procedurama koje su definisane u standardima ETS 300 324-4 i ETS 300 324-8 odnosno primenom V51DLL i V51NWKAN kojima se izvodi testiranje programskih rešenja V5.1 sprege za koje se želi potvrda o saglasnosti sa standardom ETS 300 324-1.

Testiranje je izvršeno upotrebom dva računara, tako što se na prvom računaru pokretao jedan od pomenutih programa za testiranje sa odgovarajućim argumentom kojim se identifikuje željeni test, dok je na drugom računaru bilo pokrenuto programsko rešenje V5.1 sprege koje je prikazano u ovom radu.

Testovi su uspešno sprovedeni redosledom kojim su definisani u standardima pridržavajući se odgovarajuće procedure prilikom izvođenja svakog testa posebno. U testovima kod kojih je to bilo potrebno u okviru ovog programskog rešenja pokretane su odgovarajuće funkcije koje su bile neophodne za uspešno sprovođenje određenih testova.

Centralno mesto u ovom diplomskom radu zauzima V5.1 sprega pri čemu su njene osnovne karakteristike sledeće:

- V5.1 sprega funkcioniše preko jednog linka od 2048kbit/s,
- Podržava prenos podataka za PSTN i ISDN BA korisničke priključke,
- Maksimalan broj korisničkih priključaka je 30 PSTN odnosno 15 ISDN BA,
- Korisnički priključci se statički dodeljuju odgovarajućim korisnicima kroz konfiguraciju V5.1 sprege. Ova konfiguracija može da se promeni samo postupkom ponovne konfiguracije V5.1 sprege,

Proširenje V5.1 sprege i poboljšanje navedenih karakteristika moguće je izvesti primenom V5.2 sprege koja je definisana u standardu ETS 300 347-1. V5.2 sprega je dizajnirana da bude nadogradnja V5.1 sprege a ne njena zamena ona značajno proširuje broj korisničkih priključaka i dodaje nove protokole na mrežnom nivou koji povećavaju pouzdanost celokupnog sistema.

Osnovne karakteristike V5.2 sprege su sledeće:

- V5.2 sprega funkcioniše preko jednog ili više linkova (najviše 16) od 2048kbit/s,
- Podržava prenos podataka za PSTN, ISDN BA i ISDN PRA (ISDN PRA – Integrated Services Digital Primary Rate Access).
- Maksimalan broj istovremenih komunikacionih kanala je 480,
- V5.2 sprega poseduje osobine koncentratora što znači da se korisnički priključci dinamički dodeljuju korisnicima. Posledica ove karakteristike V5.2 sprege je mogućnost priključenja do 2000 PSTN korisnika sa odnosom koncentracije 4:1
- Na mrežnom nivou dodati su sledeći protokoli:
  - **BCC protokol** (BCC – Bearer Channel Connection) kontroliše noseće kanale, pre svega se misli na govorne kanale, koji su preostali posle zauzimanja potrebnog broja signalnih kanala i privremeno ili trajno iznajmljenih linija. Na taj način se izbegava statičko zauzimanje resursa i omogućava se koncentracija korisničkih priključaka.

  - **LC protokol** (LC – Link Control) vodi računa o stanju svakog pojedinačnog linka. U slučaju otkaza ili potrebe za održavanjem linkova ovaj protokol omogućava kontolisano prevođenje iz operativnog u neoperativno stanje.

  - **PP protokol** (PP – Protection Protocol) sadrži dve zaštitne grupe koje povećavaju pouzdanost celog sistema. Prva grupa štiti signalizacione kanale u slučaju otkaza dok se drugom grupom može zaštititi bilo koji aktivni kanal. Ovaj protokol udvaja zaštićene kanale i raspoređuje ih na dva fizička linka.

7. Skraćenice

AN – Access Network
BA – Basic Access
BCC – Bearer Channel Connection
BI – Invalid Behaviour
BIOS – Basic I/O System
BO – Inopportune Behaviour
BV – Valid Behaviour
CA – Capability
CC – Common Control
CTRL – Control protokol
DDI – Direct Dialling In
DTMF – Dual Tone Multiple Frequency
EA – Extension Address
EF – Envelope Function
ET – Exchange Termination
ETS – European Telecommunication Standard
ETSI – European Telecommunications Standards Institute
FCS – Frame Check Sequence
FIFO – First In First Out
FSM – Finite state machine, konačni automat
ISA – Industry Standard Architecture
ISDN – Integrated Services Digital Network
ISO – International Organization for Standardization
IT – Basic Interconnection
ITU – International Telecommunication Union
LAPV5– Link Access Protocol for V5 interface
LC – Line Circuit
LE – Local Exchange
LT – Line Termination
MFE – Multiple Frame Established
NT1 – Network Termination 1
OSI – Open Systems Interconnection
PABX – Private Automatic Brach eXchange
PC – Port Control
PCM – Pulse Code Modulation
PP – Protection Protocol
PRA – Primary Rate Access
PSTN – Public Switched Telephone Network
Q_{AN} – Q interface at the AN
Q_{LE} – Q interface at the LE
SDL – Specification and description language
TE – Terminal Equipment
TI – Timer
TMN – Telecommunication Management Network

8. Literatura

1. ETS 300 324-1 (1994), V5.1 Interface for the support of Access Network (AN) Part 1: V5.1 Interface specification,

2. ETS 300 324-4 (2000), V5.1 Interface for the support of Access Network (AN) Part 4: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma specification for the network layer (AN side),

3. ETS 300 324-8 (1999), V5.1 Interface for the support of Access Network (AN) Part 8: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma specification for the data link layer,

4. ETS 300 125 (1991), User-network interface data link layer specification: Application of CCITT Recommendations Q.920/I.440 and Q.921/I.441

5. ETS 300 347-1 (1994), V5.2 Interface for the support of Access Network (AN) Part 1: V5.2 Interface specification,

6. Ivan Velikić: Jedno rešenje sistemskih podloga programske podrške centra za distribuciju poziva.

7. ITU-T G.964 (1994), V5.1 Interface for the support of Access Network (AN)

8. ETS 300 166 (1993), Transmission and Multiplexing (TM); Physical and electrical characteristics of hierarchical digital interfaces for equipment using the 2048kbit/s-based plesiochronous or synchronous digital hierarchies,

9. ETS 300 167 (1993), Transmission and Multiplexing (TM), Functional characteristics of 2 048 kbit/s interfaces,

10. ETS 300 376-1 (1994), Q3 interface at the Access Network (AN) for configuration management of V5 interfaces and associated user ports, Part 1: Q3 interface specification