



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Александар Зељковић

**Реализација софтвера за управљање
SPI магистралом у систему заснованом
на Texas Instruments TMS570
платформи**

МАСТЕР РАД

Нови Сад, 2015

	УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА	Број:
	21000 НОВИ САД, Трг Доситеја Обрадовића 6	
	ЗАДАТАК ЗА МАСТЕР РАД	Датум:

(Податке уноси предметни наставник - ментор)

СТУДИЈСКИ ПРОГРАМ:	Рачунарство и аутоматика
РУКОВОДИЛАЦ СТУДИЈСКОГ ПРОГРАМА:	Проф. др Никола Јорговановић

Студент:	Александар Зељковић	Број индекса:	E2 45/2014
Област:	Архитектуре и алгоритми ДСП-а 2		
Ментор:	др. Јелена Ковачевић		
НА ОСНОВУ ПОДНЕТЕ ПРИЈАВЕ, ПРИЛОЖЕНЕ ДОКУМЕНТАЦИЈЕ И ОДРЕДБИ СТАТУТА ФАКУЛТЕТА ИЗДАЈЕ СЕ ЗАДАТАК ЗА МАСТЕР РАД, СА СЛЕДЕЋИМ ЕЛЕМЕНТИМА: - проблем – тема рада; - начин решавања проблема и начин практичне провере резултата рада, ако је таква провера неопходна;			

НАСЛОВ МАСТЕР РАДА:

Реализација софтвера за управљање SPI магистралом у систему заснованом на Texas Instruments TMS570 платформи

ТЕКСТ ЗАДАТКА:

У оквиру задатка потребно је реализовати софтвер за управљање SPI магистралом у једном решењу сигурносног система за контролу у индустријском постројењу. Потребно је реализовати посебан део за слање контролних сигнала, а посебан део за слање осталих података. Приликом реализације потребно је водити рачуна о потрошњи меморије и хардверским ограничењима. С обзиром да се реализовани модул користи у сигурносном систему, обезбедити интеграцију и високу поузданост решења праћењем V развојног модела. Том приликом потребно је валидирати решење јединичним, интеграционим и системским тестовима како би се испоштовао целокупан V модел развоја система.
--

Руководилац студијског програма:	Ментор рада:



УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР:	
Идентификациони број, ИБР:	
Тип документације, ТД:	Монографска документација
Тип записа, ТЗ:	Текстуални штампани материјал
Врста рада, ВР:	Дипломски – мастер рад
Аутор, АУ:	Александар Зељковић
Ментор, МН:	др Јелена Ковачевић
Наслов рада, НР:	Реализација софтвера за управљање SPI магистралом у систему заснованом на Texas Instruments TMS570 платформи
Језик публикације, ЈП:	Српски / латиница
Језик извода, ЈИ:	Српски
Земља публикавања, ЗП:	Република Србија
Уже географско подручје, УГП:	Војводина
Година, ГО:	2015
Издавач, ИЗ:	Ауторски репринт
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6
Физички опис рада, ФО: (поглавља/страна/ цитата/табела/слика/графика/прилога)	7/46/0/11/20/0/0
Научна област, НО:	Електротехника и рачунарство
Научна дисциплина, НД:	Рачунарска техника
Предметна одредница/Кључне речи, ПО:	SPI магистрала, Texas Instruments TMS570, индустријски систем за контролу, V модел развоја
УДК	
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, ВН:	
Извод, ИЗ:	У раду је представљена реализација софтвера за управљање SPI магистралом у једном решењу сигурносног система за контролу у индустријском постројењу. Решење обезбеђује двосмерну комуникацију на две SPI магистрале, једна за слање контролних сигнала, а друга за слање управљачких података. Развој подразумева дефинисање формалних захтева, архитектуре, дизајна, имплементације и верификације према предложеном V моделу. Програмско решење је успешно интегрисано у постојећи систем, а поузданост система је верификована низом јединичних, интеграционих и системских тестова.
А.Датум прихватања теме, ДП:	
Датум одбране, ДО:	23.11.2015.
Чланови комисије, КО:	Председник: др Илија Башичевић
	Члан: др Растислав Струхарик
	Члан, ментор: др Јелена Ковачевић
	Потпис ментора



KEY WORDS DOCUMENTATION

Accession number, ANO :	
Identification number, INO :	
Document type, DT :	Monographic publication
Type of record, TR :	Textual printed material
Contents code, CC :	Master Thesis
Author, AU :	Aleksandar Zeljković
Mentor, MN :	PhD Jelena Kovačević
Title, TI :	Implementation of a SPI bus controller software based on Texas Instruments TMS570 platform
Language of text, LT :	Serbian
Language of abstract, LA :	Serbian
Country of publication, CP :	Republic of Serbia
Locality of publication, LP :	Vojvodina
Publication year, PY :	2015
Publisher, PB :	Author's reprint
Publication place, PP :	Novi Sad, Dositeja Obradovica sq. 6
Physical description, PD : (chapters/pages/ref./tables/pictures/graphs/appendixes)	7/46/0/11/20/0/0
Scientific field, SF :	Electrical Engineering
Scientific discipline, SD :	Computer Engineering, Engineering of Computer Based Systems
Subject/Key words, S/KW :	SPI bus, Texas Instruments TMS570, industry control system, V model software development
UC	
Holding data, HD :	The Library of Faculty of Technical Sciences, Novi Sad, Serbia
Note, N :	
Abstract, AB :	This master work gives a description of implementation of a SPI bus controller used in safe partition software of an industrial system. This solution enables duplex communication on two SPI buses, one is used for control signals, and second is used for management data. Development of this software requires definition of formal requirements, design architecture, implementation and verification according to V-model development process. Software is successfully integrated into existing system and reliability has been verified using unit, integration and system tests.
Accepted by the Scientific Board on, ASB :	
Defended on, DE :	23.11.2015.
Defended Board, DB :	
President:	Ilija Bašičević, PhD
Member:	Rastislav Struharik, PhD
Member, Mentor:	Jelena Kovačević, PhD

	Menthor's sign

SADRŽAJ

1. Uvod.....	1
2. Teorijske osnove	3
2.1 Sigurnosni sistem za kontrolu u industrijskom okruženju	3
2.2 Texas Instruments TMS570 platforma.....	5
2.3 SPI sprega.....	7
2.3.1 Pregled SPI sprege	7
2.3.2 Funkcionisanje SPI magistrale	9
2.3.3 Prednosti i mane SPI sprege	10
2.4 V-model razvoja softvera	11
3. Koncept rešenja.....	14
3.1 SPI drajver.....	14
3.2 Upravljanje SPI magistralom	17
3.3 SPI kontroler	21
3.3.1 Modul za prenos upravljačkih podataka	23
3.3.2 Modul za prenos kontrolnih podataka	25
4. Programsko rešenje.....	27
4.1 Pregled implementacije	27
4.2 Strukture podataka korišćenih u realizaciji	31
4.3 Opis implementiranih funkcija.....	34
4.3.1 spi_bridge_init()	34
4.3.2 spi_bridge_task()	34
4.3.3 cd_spi_bridge_get_stats()	34
4.3.4 md_spi_bridge_get_stats().....	35

4.3.5	spi_send_data()	35
4.3.6	cd_spi_bridge()	36
4.3.7	md_spi_bridge()	36
5.	Ispitivanje i verifikacija	38
5.1	Jedinično testiranje SPI kontrolera	38
5.2	Integraciono testiranje SPI kontrolera	40
5.3	Sistemska testiranje SPI kontrolera	42
6.	Zaključak	45
7.	Literatura	46

SPISAK SLIKA

<i>Slika 1. Blok dijagram jednog industrijskog sistema za kontrolu</i>	<i>4</i>
<i>Slika 2. Funkcionalni blok dijagram TMS570 sistema</i>	<i>6</i>
<i>Slika 3. Tok signala u SPI sprezi.....</i>	<i>8</i>
<i>Slika 4. Šema V modela razvoja softvera</i>	<i>12</i>
<i>Slika 5. Primer različitih nivoa zahteva i verifikacije.....</i>	<i>13</i>
<i>Slika 6. Struktura SPI drajver modula</i>	<i>15</i>
<i>Slika 7. Dijagram prenosa podataka</i>	<i>16</i>
<i>Slika 8. SPI raspoređivač prenosa</i>	<i>18</i>
<i>Slika 9. Trigerovanje SPI prenosa</i>	<i>19</i>
<i>Slika 10. Izgled klase SPI raspoređivača prenosa</i>	<i>19</i>
<i>Slika 11. Izgled objekta SPI raspoređivača prenosa</i>	<i>20</i>
<i>Slika 12. Izgled klase SPI kontrolera</i>	<i>22</i>
<i>Slika 13. Izgled modula za prenos upravljačkih podataka</i>	<i>23</i>
<i>Slika 14. Izgled modula za prenos kontrolnih podataka</i>	<i>25</i>
<i>Slika 15. Aktivnost modula za prenos kontrolnih podataka</i>	<i>28</i>
<i>Slika 16. Aktivnost modula za prenos upravljačkih podataka</i>	<i>29</i>
<i>Slika 17. Primer toka podataka u industrijskom sistemu sa različitim komunikacionim spregama</i>	<i>30</i>
<i>Slika 18. Primer testne skripte za testiranje SPI modula</i>	<i>39</i>
<i>Slika 19. Opseg integracionog testiranja.....</i>	<i>41</i>
<i>Slika 20. Koraci u sistemskom testiranju</i>	<i>42</i>

SPISAK TABELA

<i>Tabela 1. Struktura okvira za prenos kontrolnih podataka</i>	<i>31</i>
<i>Tabela 2. Struktura okvira za prenos upravljačkih podataka.....</i>	<i>32</i>
<i>Tabela 3. Struktura sa podacima transmisione statistike prenosa upravljačkih podataka .</i>	<i>33</i>
<i>Tabela 4. Struktura sa podacima o okvirima kontrolnih podataka</i>	<i>33</i>
<i>Tabela 5. Struktura spi_bridge_init funkcije</i>	<i>34</i>
<i>Tabela 6. Struktura spi_bridge_task funkcije</i>	<i>34</i>
<i>Tabela 7. Struktura cd_spi_bridge_get_stats funkcije.....</i>	<i>34</i>
<i>Tabela 8. Struktura md_spi_bridge_get_stats funkcije.....</i>	<i>35</i>
<i>Tabela 9. Struktura spi_send_data funkcije.....</i>	<i>35</i>
<i>Tabela 10. Struktura cd_spi_bridge funkcije</i>	<i>36</i>
<i>Tabela 11. Struktura md_spi_bridge funkcije.....</i>	<i>37</i>
<i>Tabela 12. Najduže vreme izvršavanja funkcija</i>	<i>40</i>

SKRAĆENICE

CMOS	- <i>Complementary Metal-Oxide Semiconductor</i> , Komplementarni metal- oksid poluprovodnik
LCD	- <i>Liquid Crystal Display</i> , Displej sa tečnim kristalom
LED	- <i>Light Emmiting Diode</i> , Dioda sa svetlosnom emisijom
RAM	- <i>Random Access Memory</i> , Memorija sa nasumičnim pristupom
SCP	- <i>Safe Controller Platform</i> , Platforma sigurnosnog kontrolera
SPI	- <i>Serial Peripheral Interface</i> , Serijska periferna sprega
SSI	- <i>Synchronous Serial Interface</i> , Sinhronizovana serijska sprega
UART	- <i>Universal Asynchronius Receiver/Transmitter</i> , Univerzalni asinhroni prijemnik/odašiljač

1. Uvod

U ovom radu biće opisana realizacija dela sistema koji se koristi u industrijskim postrojenjima i koji je zadužen za kontrolu parametara za bezbednost sistema, kao i prikupljanje i obrada datih podataka. Ovo rešenje se zasniva na SPI sprezi dok kao hardversku podlogu koristi Texas Instruments TMS570 ploču.

Poslednjih decenija došlo je do velikog napretka u razvoju integrisanih kola, procesora, računara kao i čitavih računarskih sistema i softvera koji upravlja navedenim sistemima. Danas gotovo da nema oblasti u kojoj ne možemo da nađemo neko digitalno kolo koje vrši neku funkciju, od sistema za zabavu i razonodu kao što su to igračke konzole, televizori, muzički uređaji itd, preko stvari i uređaja koje koristimo svakodnevno kao što su to automobili, telefoni i razni uređaji po kući, pa sve do uređaja i sistema koji se brinu za bezbednost i ljudske živote kao što su to sistemi zaduženi za sigurnost u saobraćaju ili razni medicinski uređaji i pomagala.

U skladu sa gore navedenim, razni digitalni sistemi su počeli da se integrišu u neke čisto mehaničke sklopove za koje do pre samo deceniju, dve nije bilo zamislivo da bi trebalo ili bilo moguće da imaju bilo kakvu komplikovaniju elektroniku. Dugo nije bilo uslova za integraciju digitalne elektronike u industrijska postrojenja, pre svega usled teških uslova u kojima takvi sistemi rade, kao što su ekstremno visoke ili niske temperature, velika vlažnost, vibracije itd. Problem koji je postojao prilikom razvoja ovakvih sistema jeste nepouzdanost ovakvih komponenti usled navedenih teških uslova rada i činjenica da su industrijski sistemi zahtevali veću pouzdanost u odnosu na većinu sistema u koje se u to vreme integrisala, ili su sistemi sa odgovarajućim nivoom pouzdanosti bili cenovno neprihvatljivi. Međutim, naglim razvojem mikroelektronike ubrzo su se stvorili uslovi za proizvodnju digitalnih sistema za upotrebu u industrijskim postrojenjima koji su posedovali odgovarajući nivo pouzdanosti. Ovo je otvorilo prostora za kreiranje digitalnih sistema koji će biti u mogućnosti da kontrolišu razne parametre

sistema, vode računa o bezbednosti sistema ili radnika, olakšavaju upravljanje i odmenjuju ljude u operacijama koje su teško izvodljive ili nebezbedne za čoveka.

U drugom poglavlju rada dat je pregled teorijskih osnova, koji su bili polazna tačka pri implementaciji sistema. To podrazumeva opis sistema u kom je implementirano realizovano rešenje, opis hardverske platforme kao i opis SPI standard za prenos podataka. U trećem poglavlju dat je pregled gradivnih celina rešenja, njihove međusobne interakcije i funkcionisanja. Detalji implementacije i opis realizacije na nižem nivou dat je u četvrtom poglavlju. Peto poglavlje donosi pregled rezultata testiranja kao i opis primenjenih metoda i alata. Šesto poglavlje sadrži diskusiju karakteristika, performansi i pouzdanosti realizovanog softver, dok sedmo poglavlje daje pregled korišćene literature.

2. Teorijske osnove

U ovom poglavlju dat je pregled teorijskih osnova, koji su bili polazna tačka pri implementaciji sistema. Opisana je Texas Instruments TMS570 platforma na kojoj se zasniva ovo rešenje i okvirno je predstavljen industrijski sistem u koje je integrisano. Takođe su date i teorijske osnove SPI magistrale koje su bile potrebne za razumevanje i realizaciju zadatka.

2.1 Sigurnosni sistem za kontrolu u industrijskom okruženju

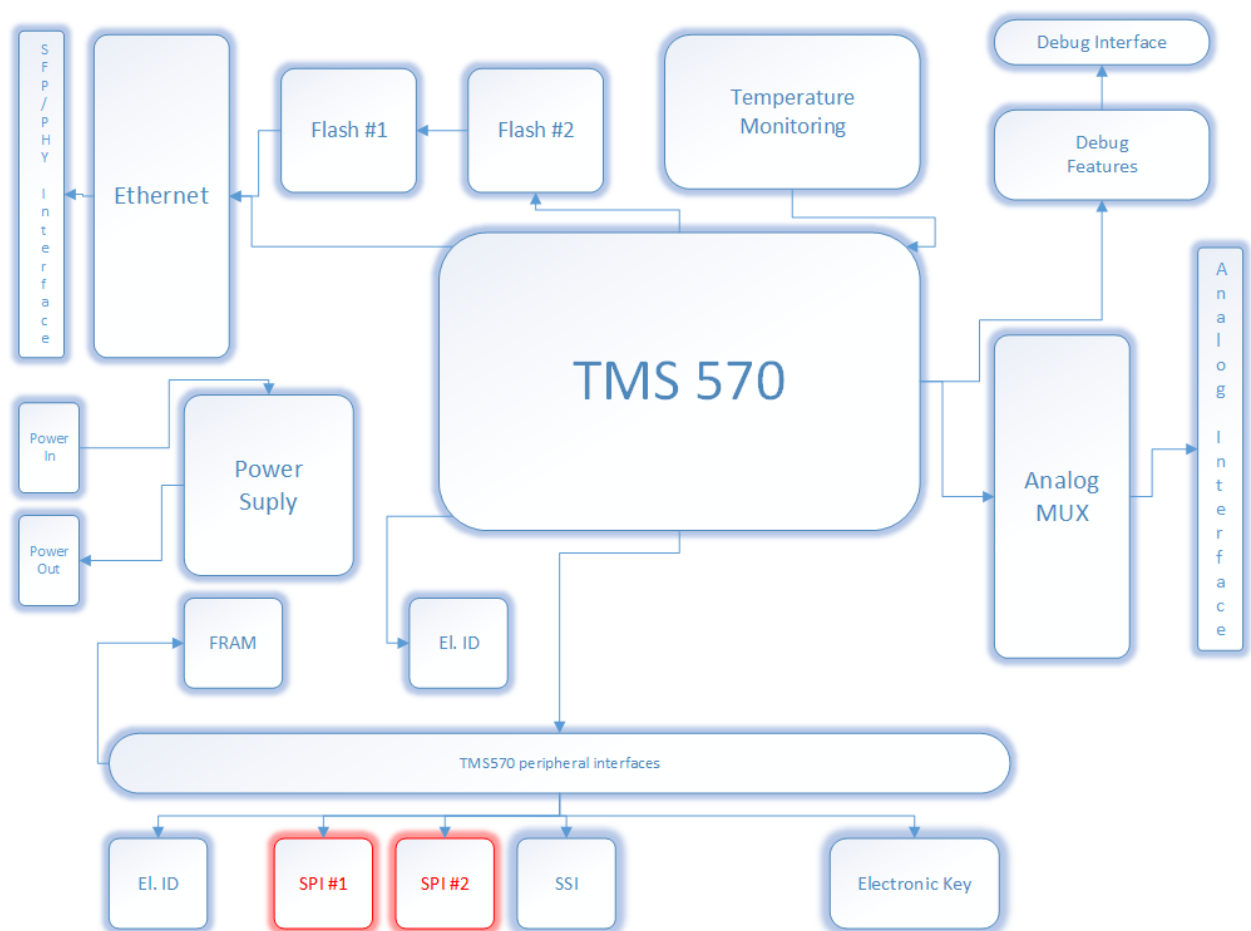
Kako bismo što bolje razumeli funkcionisanje rešenja opisanog u ovom radu, potrebno je da se upoznamo sa sistemom u koji implementiramo dato rešenje. Dati sistem sadrži nekoliko faza ulaza i izlaza kako bi obezbedio pouzdanu upotrebu senzora, aktuatora, itd. Sistem sadrži digitalne ulaze, digitalne izlaze za signale male i velike snage, analogne ulaze, SSI spregu i slično. Za potrebe komunikacije i razmene informacija (sa drugim sistemima) se koristi ethernet magistrala, dok se za dijagnostiku i monitoring koristi SPI magistrala. Sistem za kontrolu i nadzor ima dva zadatka, prvi je da obezbedi sigurnost celokupnog sistema, a drugi je da poseduje visok nivo raspoloživosti i niske troškove održavanja [1].

Svi delovi sistema su realizovani kao posebni hardverski moduli. Svrha ovakve podele je jednostavnija analiza prisustva greške u sistemu. Ovi moduli su odvojeni jasnim granicama sa ciljem sprečavanja prostiranja greške.

Sistem poseduje sledeće module:

- SCP
- SCP napajanje
- LED

- glavno napajanje
- digitalni ulazi
- analogni ulazi
- digitalni izlazi
- SSI sprega
- SPI sprega
- RS485 sprega
- SFP
- modul za elektronsku identifikaciju
- elektronski ključ
- akcelerometar



Slika 1. Blok dijagram jednog industrijskog sistema za kontrolu

Na slici 1 je dat pojednostavljeni blok dijagram industrijskog sistema za kontrolu i bezbednost, a u nastavku će biti ukratko objašnjene funkcije pojedinih njegovih komponenti.

SCP – platforma zadužena za kontrolu i nadzor sistema

SCP napajanje – ovaj modul obezbeđuje neophodno napajanje SCP modula i obezbeđuje ranu detekciju greške kao i privremeno smeštanje za druge module

LED – diode zadužene za indikaciju glavnih događaja u sistemu

glavno napajanje – vrši prilagođavanje napona i vrši napajanje celog sistema

digitalni ulazi – sistem poseduje dva tipa digitalnih ulaza:

- 0-24V za logičke signale
- 0-24V za logičke signale sa mogućnošću brojanja

digitalni izlazi - sistem poseduje četiri tipa digitalnih izlaza:

- izlaz niske snage – do 300mA
- uskladišten izlaz niske snage – do 300mA
- izlaz visoke snage – do 4,6A
- pulsirajući izlaz sa modulacijom – do 2,5A

analogni ulazi - sistem poseduje tri tipa analognih ulaza:

- strujni ulaz 4-20mA
- diferencijalni naponski ulaz 2,5V
- diferencijalni naponski ulaz sa dodatnom funkcijom dekodera 2,5V

modul za elektronsku identifikaciju – Atmel ATSHA04 čip zadužen za skladištenje relevantnih informacija i tajnih ključeva koji se koriste za proveru identiteta hardvera

elektronski ključ – ova sprega služi kako bi se na njega prikačila eksterna EEPROM memorija koja dolazi u formi ključa

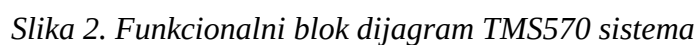
akcelerometar – hardver za merenje ubrzanja kod pokretnih delova sistema kod kojih je ova informacija neophodna

2.2 Texas Instruments TMS570 platforma

TMS570 je mikrokontroler visokih performansi koji se koristi za ugradnju u sigurnosne sisteme korišćene u industriji. TMS570 sadrži ARM Cortex R4 procesor. Ovaj procesor podržava Big-Endian (BE32) format. Platforma poseduje 384+256KB integrisane flash memorije i 32KB RAM memorije. Flash i RAM memorija poseduju detekciju i korekciju greške.

Uređaj poseduje sledeće komunikacione sprege:

- jedna MibSPI sprega
- dve SPI sprege



Zahvaljujući integrisanim podsistemima zaduženim za bezbednost i velikom broju komunikacionih i kontrolnih periferija TMS570 je sasvim pogodno rešenje kada se javi potreba za izvršavanjem sigurnosnih aplikacija za rad u realnom vremenu.

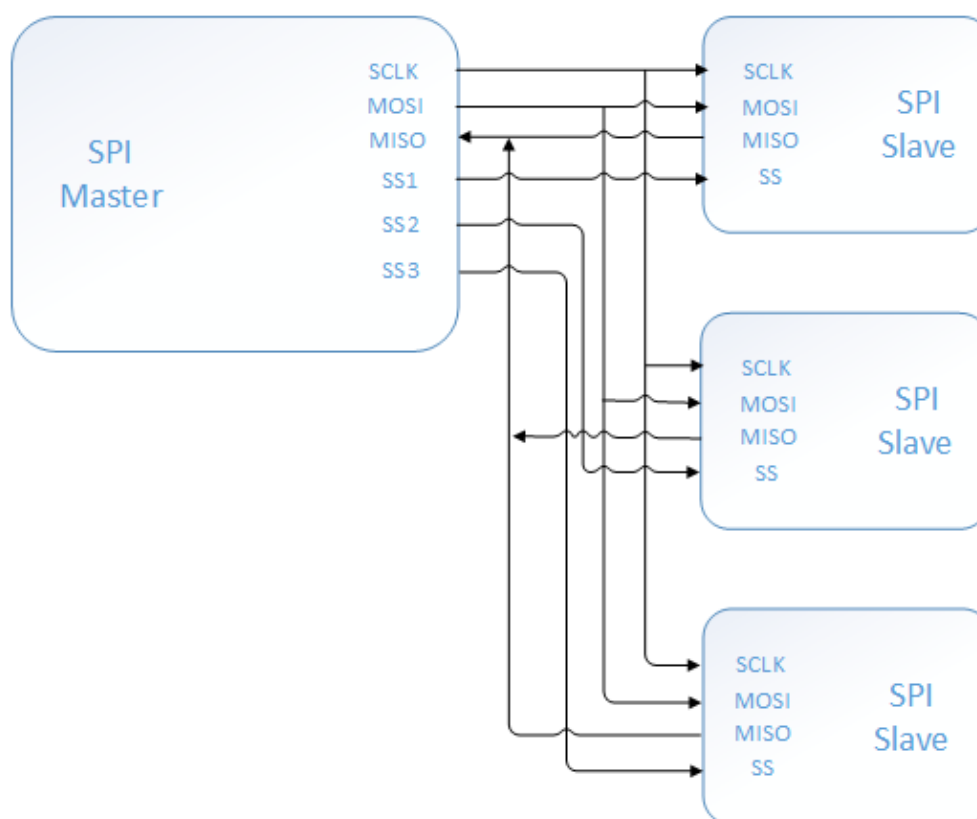
2.3 SPI sprega

2.3.1 Pregled SPI sprege

SPI je standard za sinhroni serijski prenos podataka koji se koristi za komunikaciju na manjim udaljenostima, pre svega u sistemima za ugradnju (*embedded systems*). Spregu je za svoje potrebe kreirala kompanija Motorola, za spajanje mikrokontrolera sa periferijama (vrlo slično I2C standardu). Primeri primene SPI sprege su npr. senzor kartica i LCD monitori.

SPI je još poznat i pod nazivom četvorožična veza jer u komunikaciji učestvuju četiri signala. SPI može da ostvari dvosmernu, dupleks vezu između jednog master (glavnog) uređaja i jednog ili više slave (sporednih) uređaja.[3] SPI može da se posmatra i kao serijsku sinhronu spregu, ali se razlikuje od SSI sprege koja je četvorožična simpleks sprega, i koristi drugi tip signalizacije.

Glavni uređaj koji je po pravilu mikrokontroler generiše takt (SCLK) opsega između 1MHz i 70 MHz. I glavni i sporedni uređaj imaju u sebi osmobiitni (podržane su i druge veličine) pomerački registar. Komunikacija počinje u trenutku kada glavni računar aktivira SS (slave select) signal, tj postavlja logičku nulu na datu liniju. Komunikacija je uvek dvosmerna, jedan bajt ide od glavnog prema sporednom uređaju, dok drugi bajt ide od sporednog prema glavnom uređaju. Na magistrali se konstantno odvija saobraćaj, ako mikrokontroler šalje veći broj bajtova ka periferiji, a periferija trenutno nema podataka koje bi slala mikrokontroleru, ona će slati nasumične podatke kako bi se održala komunikacija [4].



Slika 3. Tok signala u SPI sprezi

U komunikaciji između glavnog i sporednog uređaja učestvuju sledeći signali:

SCLK – Signal takta koji generiše glavni uređaj.

MOSI – *Master Output Slave Input*. Podaci od glavnih ka sporednih uređaja.

MISO – *Master Input Slave Output*. Podaci od sporednih ka glavnim uređajima.

SS – *Slave Select*. Ovaj signal generiše glavni uređaj i služi za odabir sporednog uređaja. Nula poslata na ovomvodu signalizira početak komunikacije.

Veza sa više sporednih uređaja zahteva da glavni uređaj obezbedi SS (*Slave Select*) signale za svaki sporedni uređaj. Ovakva veza zahteva i da sporedni uređaj na svom MISO izlazu poseduje kolo visoke impedanse da ne bi došlo do konflikta više izlaza koji su povezani na istu tačku. Takođe postoji mogućnost vezivanja više sporednih uređaja u lanac (*daisy chain*) kada je izlaz glavnog vezan na ulaz prvog sporednog uređaja, izlaz prvog sporednog na ulaz drugog sporednog uređaja itd. Izlaz poslednjeg sporednog uređaja je vezan na ulaz prvog glavnog uređaja.

SSI, SSP i MicroWire komunikacioni protokoli su izvedeni iz SPI protokola.

2.3.2 Funkcionisanje SPI magistrale

SPI periferija poseduje četiri registra:

- 1) SPI kontrolni registar – sadrži broj programabilnih bitova koji određuju funkciju SPI bloka. Ovaj registar je neophodno podesiti pre slanja podataka.
- 2) SPI statusni registar – sadrži (read only) bite koji služe za praćenje statusa SPI sprege. Osnovna uloga ovog registra je detekcija kraja prenosa podataka pomoću SPIF bita. Ostali biti se koriste za indikaciju specijalnih slučajeva.
- 3) SPI registar podataka sadrži podatke za prenos. Prilikom serijskog prenosa podataka u upotrebi je interni pomerački registar. Ne postoji bafer između registra podataka i internog pomeračkog registra, već se sadržaj registra podataka prebacuje direktno u interni pomerački registar zbog čega upis podataka ne treba obavljati dok je prenos u toku. Podaci za čitanje se smeštaju u bafer i kada se prenos završi primljeni podaci se prebacuju u 1-bajtni bafer podatak iz kog se kasnije čitaju.
- 4) SPI registar brojač takta – kontroliše frekvenciju kad je SPI blok u modu glavnog uređaja. Vrednost frekvencije se mora podesiti pre početka prenosa podataka. Ovaj registar nema funkciju ako je SPI blok u modu sporednog uređaja.

Ulazi i izlazi u ovoj implementaciji su standardni CMOS ulazi i izlazi. Ukoliko je uređaj u modu sporednog njegovi ulazi i izlazi će biti aktivni samo ako je njegov SSEL signal aktivan [5].

Redosled prenosa podataka na strani glavnog uređaja je sledeća:

- 1) Postavljanje željene frekvencije u registar brojač takta.
- 2) Podešavanje kontrolnog registra na željenu opciju.
- 3) Upis podataka za prenos u registar podataka i pokretanje prenosa podataka.
- 4) Čekanje da SPIF bit u statusnom registru bude postavljen na 1.
- 5) Čitanje statusnog registra.
- 6) Čitanje primljenih podataka iz registra podataka.
- 7) Za prenos dodatnih podataka ponoviti korake počev od koraka 3.

Redosled prenosa podataka na sporednoj strani je sledeća:

- 1) Podešavanje kontrolnog registra na željenu opciju.
- 2) Upisivanje podatka za upis u registar podataka.
- 3) Čekanje da SPIF bit u statusnom registru bude postavljen na 1 (nakon poslednje ivice takta prenosa podataka).
- 4) Čitanje statusnog registra.

- 5) Čitanje primljenih podataka iz registra podataka.
- 6) Za prenos dodatnih podataka ponoviti korake počev od koraka 2.

Da bi se SPIF bit postavio na 0, potrebno je pročitati ili upisati podatke u SPIF registar podataka [6].

2.3.3 Prednosti i mane SPI sprege

SPI standard poseduje određene prednosti i mane u odnosu na slične standarde, i na osnovu njih je na korisniku da izabere najpogodniji način za prenos podataka u skladu sa svojim potrebama.

Prednosti SPI sprege:

- full-duplex komunikacija
- veći protok podataka nego kod I2C magistrale
- nije ograničen na prenos 8-bitnih reči u slučaju bit prenosa
- proizvoljan izbor veličine poruke, sadržaja i namene
- jednostavan hardver
- manja potrošnje energije u odnosu na I2C usled manjeg broja kola
- sporedni uređaj koristi takt glavnog uređaja nije mu potreban precizan oscilator
- primopredajnik nije potreban
- najviše jedna jedinstvena magistrala po uređaju, ostale su deljene

Mane SPI sprege:

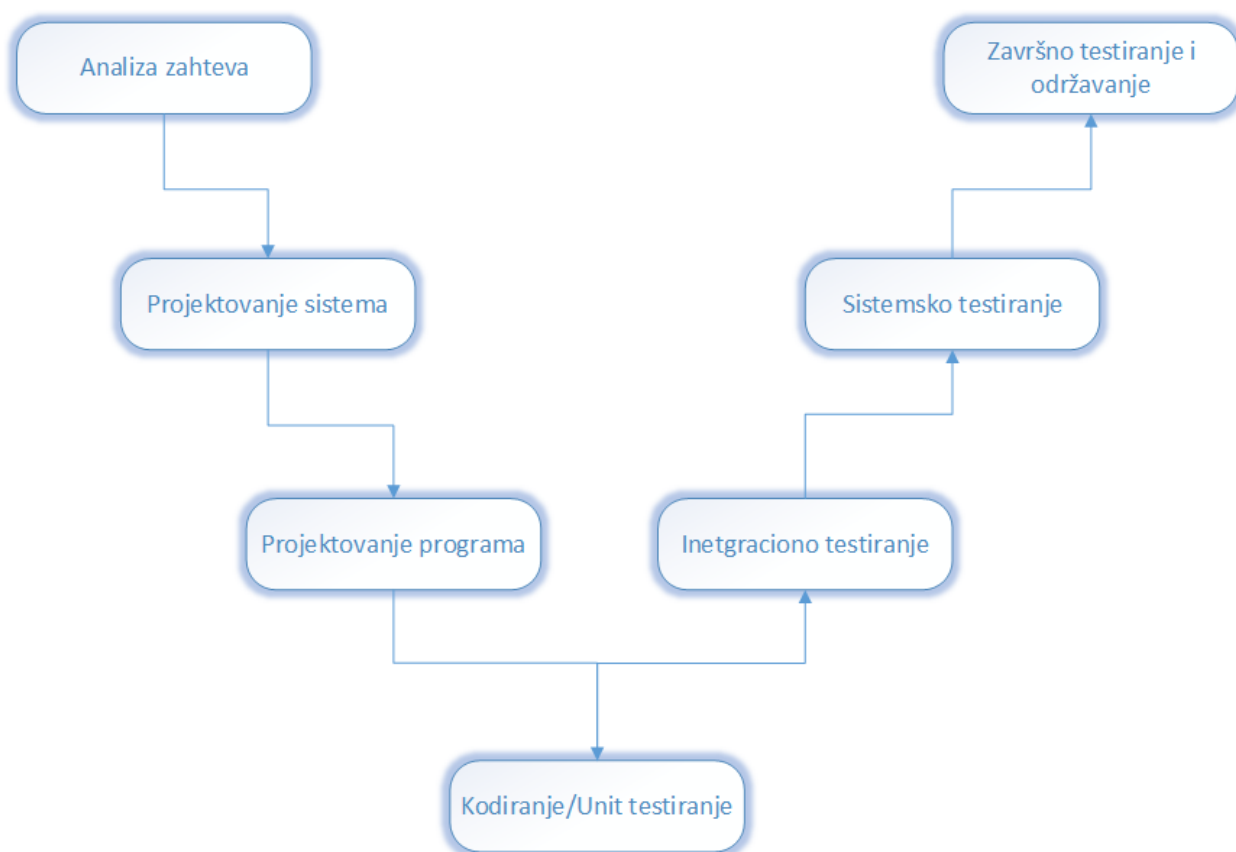
- zahteva više pinova od I2C sprege
- nema unutrašnjeg adresiranja
- nema hardverske kontrole toka podataka
- nema nikakve potvrde od strane sporednog uređaja
- sistemi sa više uređaja su retki i nezgrapni i uglavnom su ograničeni na jedan sporedni
- bez formalnog standarda, potvrda usklađenosti nije moguća
- podržava samo kratke udaljenosti za prenos podatak za razliku od RS-232, RS-485 ili CAN standarda

2.4 V-model razvoja softvera

V-model razvoja softvera predstavlja modifikovanu verziju *waterfall* modela i on prikazuje odnos aktivnosti u okviru testiranja softvera sa fazama dizajna, analize i realizacije softvera. Naziv je dobio po izgledu dijagrama koji asocira na slovo V, gde se na levoj strani nalaze faze analize i dizajna sistema, a na desnoj strani nalaze se faze testiranja i održavanja sistema, dok se na sredini nalazi faza realizacije. V-model podrazumeva testiranje delova sistema kao i testiranje pri integraciji kako bi se verifikovao dizajn programa. Tokom testiranja delova ili testiranja prilikom integrisanja ili nakon njega ljudi zaduženi za testiranje treba da obezbede da su svi aspekti programa uspešno implementirani. [7] Testiranje takođe treba da verifikuje i dizajn datog sistema kako bi se pokazalo da su svi aspekti projektovanog sistema uspešno implementirani [8].

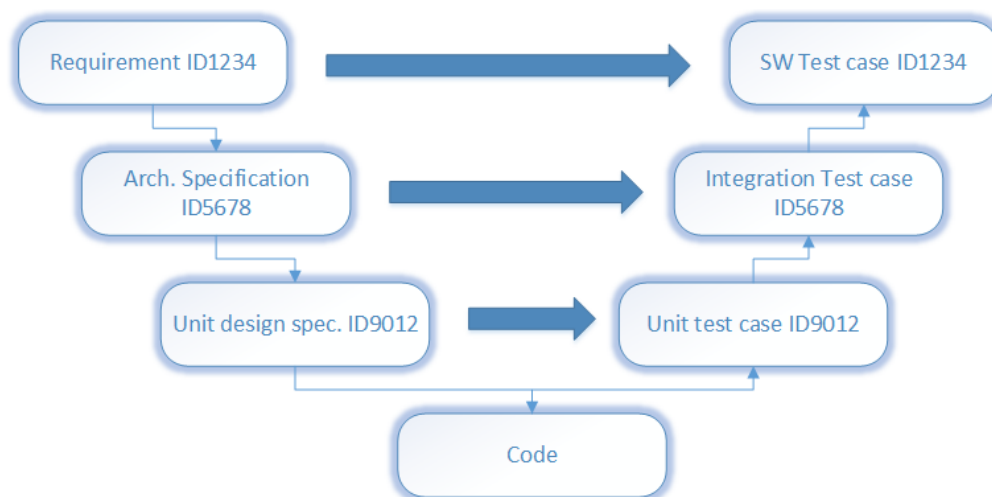
Povezanost leve i desne strane prikazuju to da problemi pronađeni tokom verifikacije i validacije moraju biti rešeni vraćanjem na odgovarajuću fazu razvoja u kojoj je dati problem i uzrokovan pre nego što se dati korak provere ponovo izvrši. V-model stavlja u fokus određene korake i prepravke koji su u modelu vodopada skriveni.

Da bi ispravno funkcionisao, sistem se testira u tri faze: testiranje pojedinačnih modula sa integracijom, testiranje integrisanog sistema i završno testiranje. Testiranje pojedinačnih modula sa integracijom i testiranje integrisanog sistema služi za verifikaciju dizajna sistema dok završno testiranje služi za validaciju sistema. Testiranje pojedinačnih modula uz postepenu integraciju (*unit* testiranje) za cilj ima da se razvojni tim uveri da je svaki aspekt sistema pravilno implementiran. Integraciono testiranje pokazuje da sistem radi kao celina, što se prema projektu od njega očekuje. Zadatak ove dve faze je da provere da li je sistem ispravno i u potpunosti implementiran prema urađenom projektu. Završno testiranje proverava da li sistem ispunjava sve zahteve navedene u specifikaciji zahteva i ovu fazu često sprovode naručioci softvera s obzirom da oni najbolje poznaju zadatak softvera [9].



Slika 4. Šema V modela razvoja softvera

Sistem za kontrolu i nadzor industrijskog sistema u koji je implementiran SPI kontroler implementiran u ovom radu je u potpunosti razvijan u skladu sa V modelom razvoja softvera. Analiza zahteva, kao najviši nivo razvoja sistema, je detaljno opisana kroz specifikaciju zahteva sistema i specifikaciju dizajna sistema, koji su bili polazne tačke pri razvoju. Ispunjenost ovih zahteva je, u skladu sa V modelom, proverena poslednja na način definisan u sistemsko-integracionoj test specifikaciji i kao rezultat je dobijen odgovarajući test izveštaj. Sledeća faza razvoja je projektovanje sistema, i njeni zahtevi su definisani u specifikaciji zahteva softvera dok je rezultat sistemskog testiranja koji je pokrio specifikaciju opisan u softver test izveštaju. Zatim se prelazi na projektovanje programa u skladu sa dokumentom koji sadrži softversku arhitekturu, da bi se kasnije kroz integraciono testiranje opisano u integracionoj test specifikaciji potvrdila ispravnost date operacije. Na samom kraju, najniža forma razvoja sistema tj. kodiranje se vrši na osnovu dizajn dokumenta, vrši se testiranje svake jedinice, a rezultat testiranja svake jedinice se dokumentuje u posebnu *unit-test* specifikaciju.



Slika 5. Primer različitih nivoa zahteva i verifikacije

Zbog svoje jednostavnosti i lake primene V model je vrlo korišćen metod za razvoj softvera. Njegove prednosti u odnosu na ostale metode su:

- *podržava povratne sprege* – V model se lako može primeniti na razvoj jednostavnih ali i složenih sistema jer dopušta vraćanje na neku od prethodnih faza razvoja sistema
- *omogućava verifikaciju i validaciju sistema* – omogućava proveru kako ispravne implementacije, tako i ispunjenost zahteva korisnika
- *obežbeđuje kvalitetan proizvod* – strogo kontrolisani model razvoja garantuje odličan kvalitet naručenog softvera
- *testiranje počinje već u ranim fazama razvoja projekta* – testiranje kreće u samom startu razvoja projekta i kreće se kroz celokupan proces razvoja što značajno smanjuje vreme ispravljanja grešaka a samim tim i troškove izrade sistema

Mane V modela su:

- *mala fleksibilnost* – u slučaju uočene greške u ranijoj fazi i rešavanju iste potrebno je ponoviti i sve naredne faze kako bi se uočile eventualne novonastale greške
- *zahteva velike resurse* – realizacija sistema po ovom modelu zahteva dosta velika novčana sredstva tako da je najpogodniji za veće projekte odnosno kompanije

3. Koncept rešenja

U ovom poglavlju bliže je opisan SPI modul koji je realizovan, sa osvrtom na njegovu vezu sa celokupnim sistemom u koji je integrisan.

3.1 SPI drajver

Glavni zadatak SPI drajvera je slanje i prijem podataka koji će se prenositi ovom magistralom na TMS570 okruženju. Ovaj modul će biti implementiran kao deo sloja za apstrakciju hardvera kako bi se lakše integrisao sa ethernet softverskom platformom.

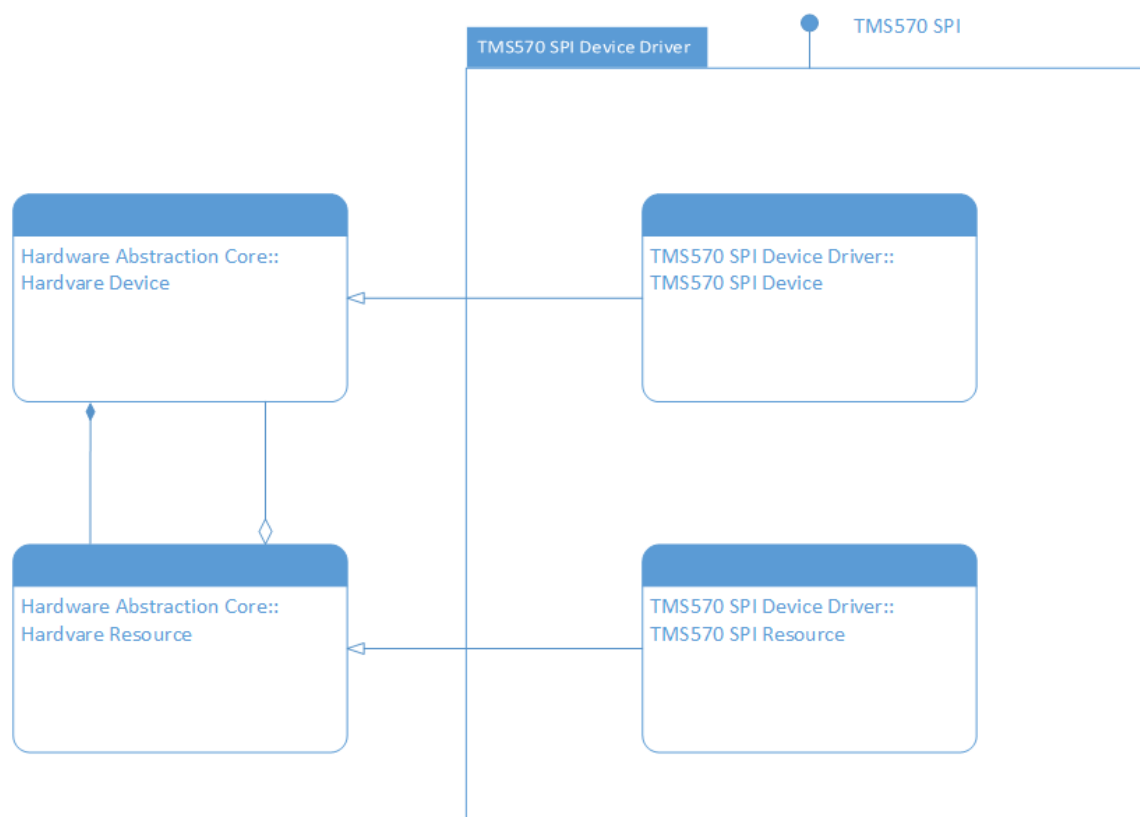
Neophodno je da SPI drajver podržava sledeće funkcije:

- 1) Atomski prenos delova (chunk) podataka – ova mogućnost je neophodna kako prenos podataka ne bi ometao neki drugi softver.
- 2) Kompatibilnost i multi-buffer mod – TMS570 daje mogućnost za korišćenje takozvanog multi-buffer moda koji omogućava korisniku čuvanje podataka u posebnoj memoriji pre slanja.

Kao što se može videti na slici 4, TMS570 SPI drajver modul se sastoji iz dva dela. Prvi je TMS570 uređaj, klasa koja nasleđuje klasu jezgra za apstrakciju hardvera i koja sprovodi rukovanje različitim periferijama koje se povezuju na odgovarajuću SPI magistralu. Drugi je TMS570 SPI resurs klasa koja nasleđuje klasu jezgra za apstrakciju hardvera zaduženu za rukovanje resursima i implementira virtuelne funkcije definisane u svojoj osnovnoj klasi kao i funkcije za atomski prenos podataka. Za slanje i prijem podataka preko SPI magistrale koristiće se posebna sprega.

Klasa TMS570 uređaja obezbeđuje čitanje podatka čiju veličinu bloka određuje TMS570 klasa za rukovanje resursima putem I/O kontrolnih komandi. Ova informacija je potrebna višim

nivoima softvera kako bi se znalo da se podaci šalju blokovski, kao i veličina bloka koja će da se šalje. U slučaju moda sa više bafera, takođe je potrebno znati i veličinu čitave grupe podataka koja se šalje tokom jedne sesije.



Slika 6. Struktura SPI drajver modula

Prilikom slanja podataka, SPI magistralom se neće slati samo korisni podaci, već i razni dodatni podaci kao što su adrese, zaglavlja itd. TMS570 SPI uređaj ne zna koja je struktura ovih zaglavlja. Zbog toga, prilikom prenosa velikih blokova podataka, iste je potrebno slati u vidu podblokova podataka odgovarajuće veličine i sa odgovarajućim zaglavljima. SPI drajver ne zna strukturu ovih zaglavlja tako da bi slanje malih podblokova dovelo do oštećenja podataka. Stoga kontroler fleš memorije mora da obezbedi slanje velikih blokova podeljenih u manje podbloke određene veličine koji će se jedan za drugim u jednoj sesiji slati SPI magistralom. Klasa TMS570 SPI uređaja mora biti u stanju da obradi dovoljno instanci klase resursa, tj onoliko instanci koliko čipova bude zahtevalo saobraćaj preko magistrale (propusnost magistrale će odrediti koliko uređaja može da je koristi).

Klasa resursa bi trebalo da podržava:

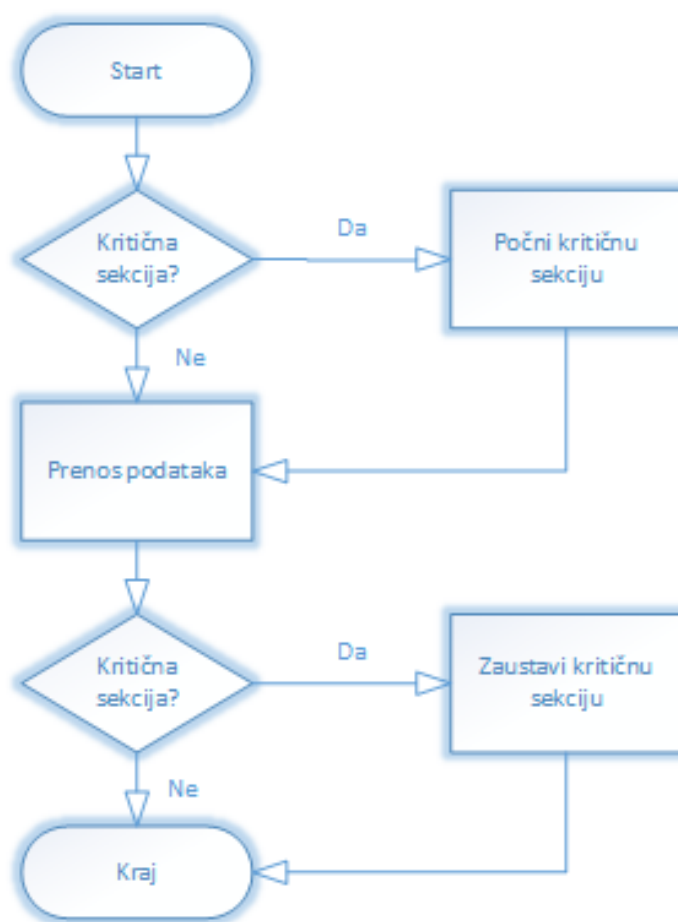
- pozivanje resursa
- asinhroni prenos podataka
- direktan pristup resursima

Takođe klasa resursa implementira virtuelne funkcije modula za apstrakciju hardvera tj klase za sinhroni (asinhroni prenos nije podržan) upis i čitanje podataka sledeće granulacije:

- 8 bita
- 16 bita
- 32 bita
- bloka bajtova

TMS570 klasa konfiguracije resursa je izvedena iz klase konfiguracije resursa modula za apstrakciju hardvera i ona mora da sadrži sledeća polja:

- konfiguracija prenosa koja se koristi za komunikaciju sa resursima
- konfiguracija transfera koja se koristi ako je modul konfigurisan da sa resursima komunicira u multi-buffer modu
- veličina prenosnog bloka koja se koristi ako je modul u normalnom režimu prilikom komunikacije sa resursima
- referenca na funkciju za startovanje kritične sekcije
- referenca na funkciju za zaustavljanje kritične sekcije



Slika 7. Dijagram prenosa podataka

Kao što je prikazano na slici 5, operacija prenosa podataka počinje pokretanjem kritične sekcije, ali samo ako je ona zahtevana od strane korisnika (kritična sekcija je definisani vremenski interval u kom zadatak koji se trenutno izvršava ne može da bude prekinut nekim drugim zadatkom u sistemu). Zatim sledi prenos podataka, bio to upis ili čitanje podataka. Na kraju se zaustavlja kritična sekcija. Ova procedura se koristi bez obzira na to kako je konfigurisan TMS570 SPI modul. Postoje dve definisane *callback* funkcije, jedna za pokretanje i druga za zaustavljanje kritične sekcije. Kritične sekcije bi trebalo da koriste sve funkcije koje koriste SPI magistralu.

3.2 Upravljanje SPI magistralom

Sledeći uređaji koriste SPI magistralu na TMS570 uređaju:

- uređaj za elektronsku identifikaciju
- elektronski ključ
- SSI senzor x 3
- akcelerometar

Za uređaj za elektronsku identifikaciju i elektronski ključ iščitavaju podatke svakih 0,5ms (20 puta u 10ms) dok SSI senzori i akcelerometar očitavaju podatke svakih 10ms. Svi transferi podataka moraju da se odvijaju bez intervencije centralnog procesora. U ovu svrhu se koristi *multi-buffer* mod, dok procesor samo inicira transfer podataka svakih 10ms.

Prilikom upravljanja SPI magistralom koriste se sledeći moduli:

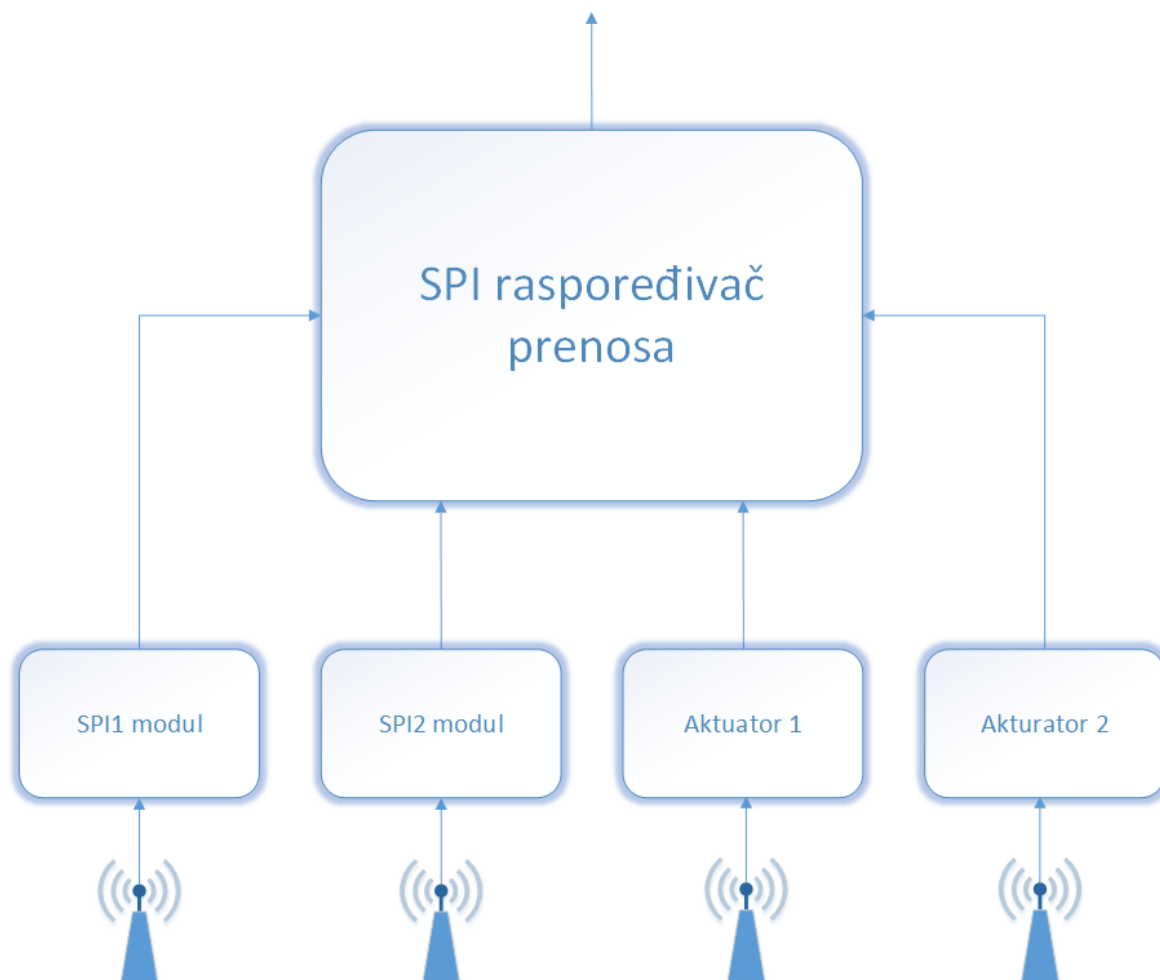
- SPI raspoređivač prenosa – vrši raspoređivanje prenosa po SPI uređajima u okviru jedne transfer grupe
- SPI raspoređivač vremena – kontroliše ugrađeni SPI tajmer

Multi-buffer mod SPI magistrale koristi transfer grupe za različite uređaje. Postoje tri transfer grupe koje zauzimaju sledeću količinu memorije:

- uređaj za elektronsku identifikaciju - 15 bajta
- elektronski ključ – 13 bajta
- akcelerometar – 10 bajta

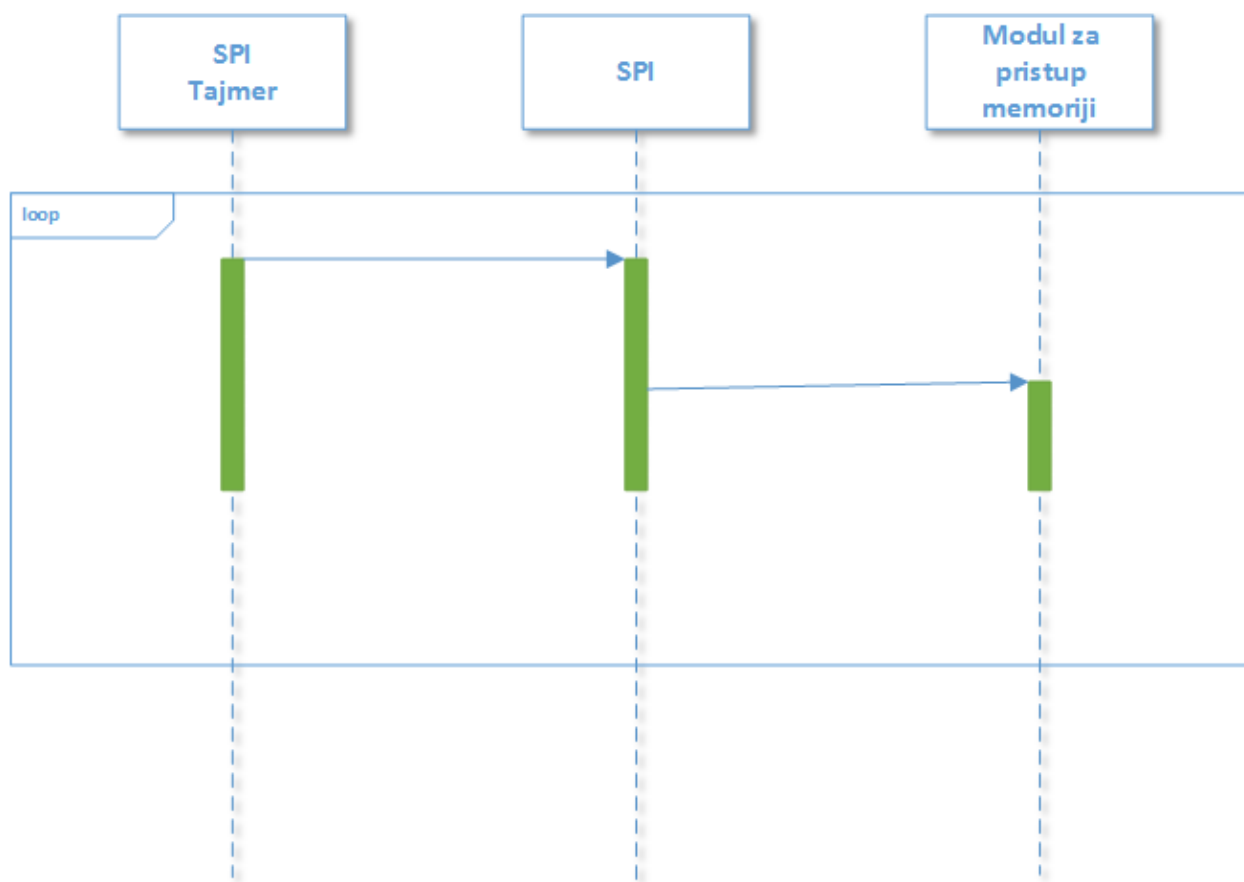
Svim transfer grupama upravlja SPI tajmer. Ovaj tajmer se postavlja na određeni vremenski interval, i kada vremenski interval istekne inicira se prenos podataka.

Transfer grupa 3 (akcelerometar) se deli između više uređaja s obzirom da se podaci ne čitaju često. Zbog toga ova grupa može da podrži istovremeno više uređaja tog tipa. Da bi ovakav prenos podataka mogao da bude realizovan potrebno je uključiti i SPI raspoređivač prenosa.



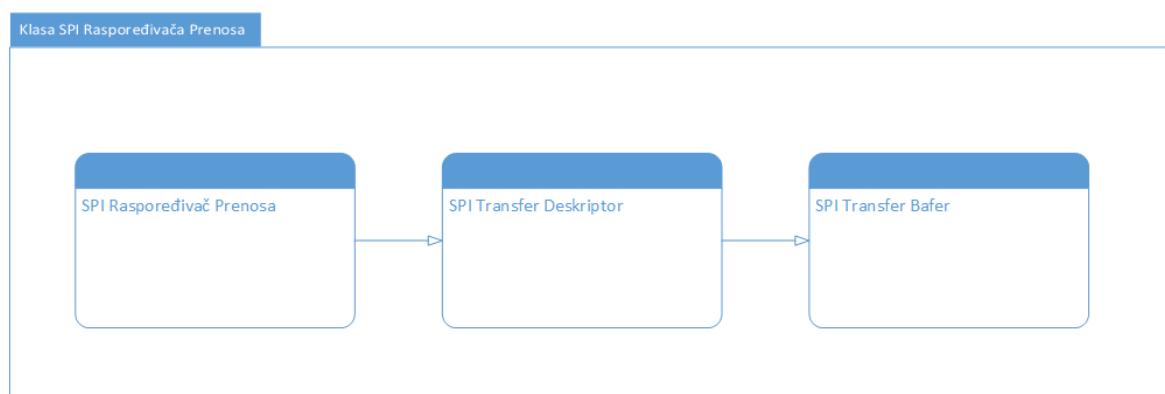
Slika 8. SPI raspoređivač prenosa

SPI tajmer je podešen tako da se aktivira 20 puta u 10ms, tako da se on pokreće 20 puta u jednoj periodi aplikacije. SPI raspoređivač prenosa pokreće transfer samo 4 puta i onda stane. Transfer grupa 2 se koristi za 4 SPI uređaja gde je podatke sa jednog potrebno očitati samo jednom za vreme jedne periode aplikacije. Kada SPI tajmer pošalje okidački signal, sve tri transfer grupe su pokrenute. Postoji nivo prioriteta kod transfer grupa koji određuje kojim redosledom će da se prenose podaci, transfer grupe poseduju jedinstveni identifikacioni broj, gde transfer grupa sa najmanjim identifikacionim brojem ima najveći prioritet.



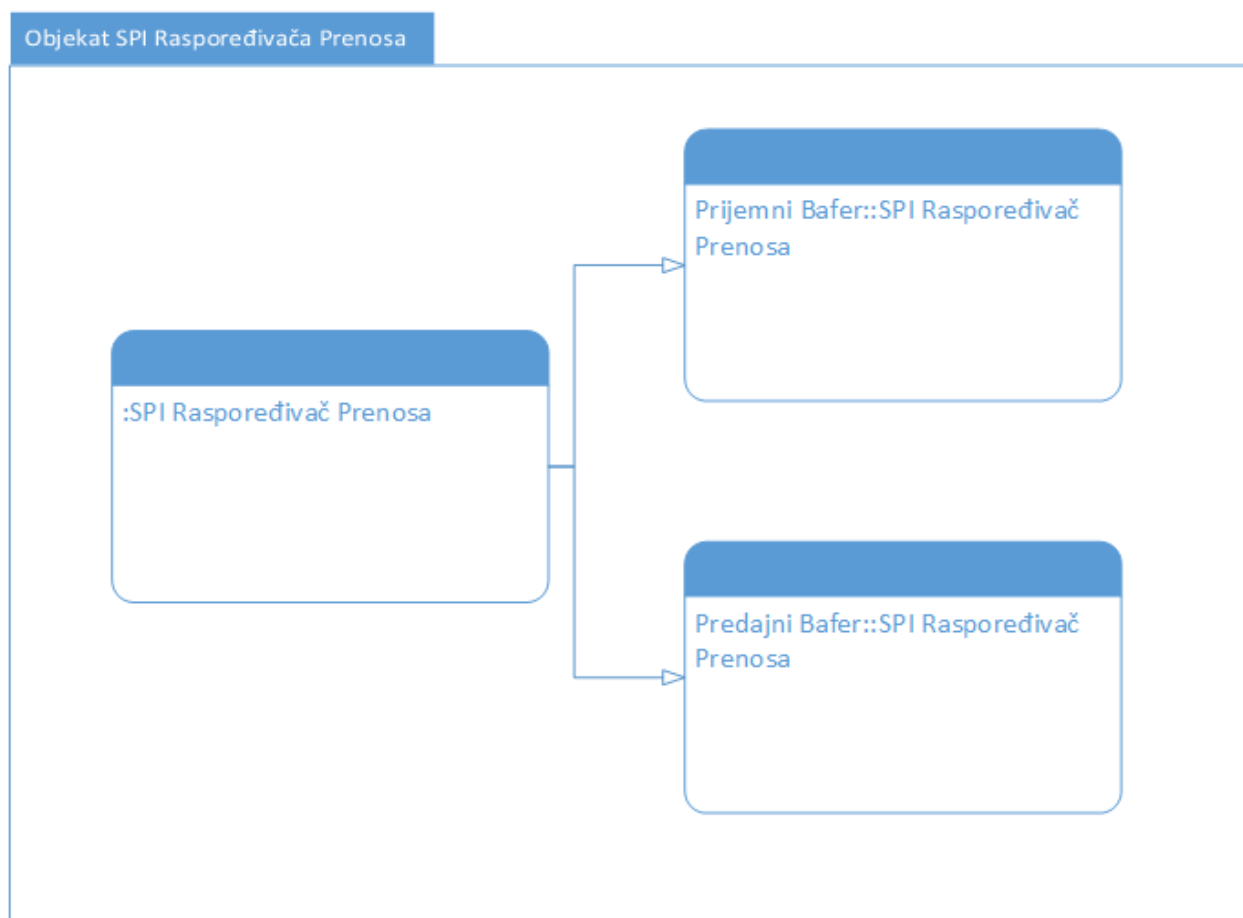
Slika 9. Trigerovanje SPI prenosa

SPI raspoređivač prenosa definiše deskriptor transfera podataka za prenos SPI magistralom u multi-buffer modu koristeći modul za pristup memoriji. On obezbeđuje razmenu podataka između transfer grupa svakim okidačem modula za pristup memoriji. Ovo omogućava upotrebu transfer grupa za višestruki prenos podataka između različitih SPI uređaja.



Slika 10. Izgled klase SPI raspoređivača prenosa

SPI raspoređivač je bafer logički podeljen na nekoliko deskriptora, gde svaki deskriptor definiše prenos jednog SPI uređaja. Deskriptor sadrži više SPI prenosnih bafera koji odgovaraju baferi SPI transfer grupe i sadrži 16-bitne kontrolne reči i 16-bitne reči podataka. Kontrolna reč definiše parametre transfera dok reč podataka sadrži podatke za slanje. Svaki deskriptor sadrži isti broj prenosnih bafera jer veličina korišćene transfer grupe treba da bude nameštena nakon inicijalizacije.



Slika 11. Izgled objekta SPI raspoređivača prenosa

Raspoređivač se sastoji iz dva dela, bafera za slanje podataka na SPI magistralu i bafera za prijem podataka sa SPI magistrale. U baferu za prijem podataka kontrolna reč ima značenje statusa transfera podataka.

Raspoređivač prenosa poseduje sledeće konfiguracione parametre:

- konfiguracija trenutno korišćene SPI jedinice
- broj bafera u jednom deskriptoru
- broj deskriptora u jednom raspoređivaču

- identifikacioni broj kanala modula za pristup memoriji koji se koristi za slanje na SPI magistralu
- identifikacioni broj kanala modula za pristup memoriji koji se koristi za prijem sa SPI magistrale
- identifikacioni broj transfer grupe
- bafer za slanje podataka
- bafer za primanje podataka

Ovi parametri su neophodni raspoređivaču prenosa zbog njegovog ispravnog funkcionisanja i oni će biti prosleđeni raspoređivaču kao ulazni parametar. Raspoređivač treba da obezbedi funkcionalnost inicijalizacije modula za pristup memoriji zbog kopiranja podataka između SPI transfer grupe i bafera kao i funkcionalnost prenosa podataka iz prijemnog u odredišni bafer. Iz svakog prenosnog bafera potrebno je kopirati samo niže bite podatka, pošto mu je namena prenos osmobičnih podataka preko SPI magistrale. Za korišćenje raspoređivača prenosa potrebno je da podaci u baferu ne sadrže dodatne statusne informacije.

Raspoređivač prenosa koristi sledeće module:

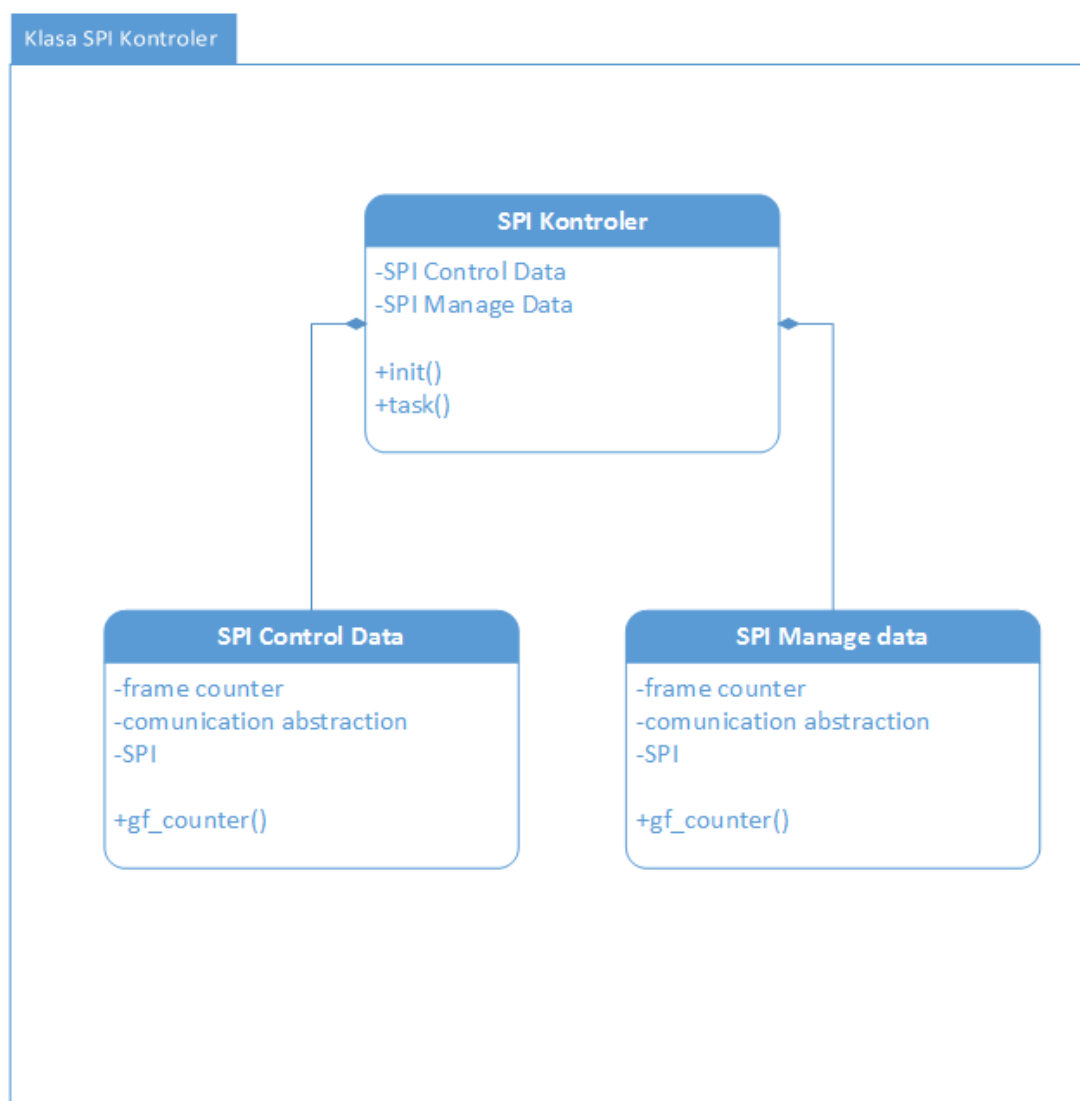
- SPI drajver za pristup SPI magistrali
- modul za pristup memoriji

SPI tajmer kontroliše ugrađeni tajmer SPI magistrale. On se koristi kako bi povremeno pokrenuo preuzimanje podataka sa SSI senzora i aktuatora. SPI tajmer takođe poseduje mogućnost da inicira transfer modula za pristup memoriji. Neophodno je da tajmer poseduje funkcionalnost resetovanja na osnovnu početnu vrednost.

3.3 SPI kontroler

Slanje okvira sa podacima se vrši preko SPI magistrale pomoću SPI kontrolera. Da bi se razdvojio prenos kontrolnih podataka i prenos ostalih podataka SPI kontroler je podeljen u dva modula:

- modul za prenos kontrolnih podataka
- modul za prenos upravljačkih podataka



Slika 12. Izgled klase SPI kontrolera

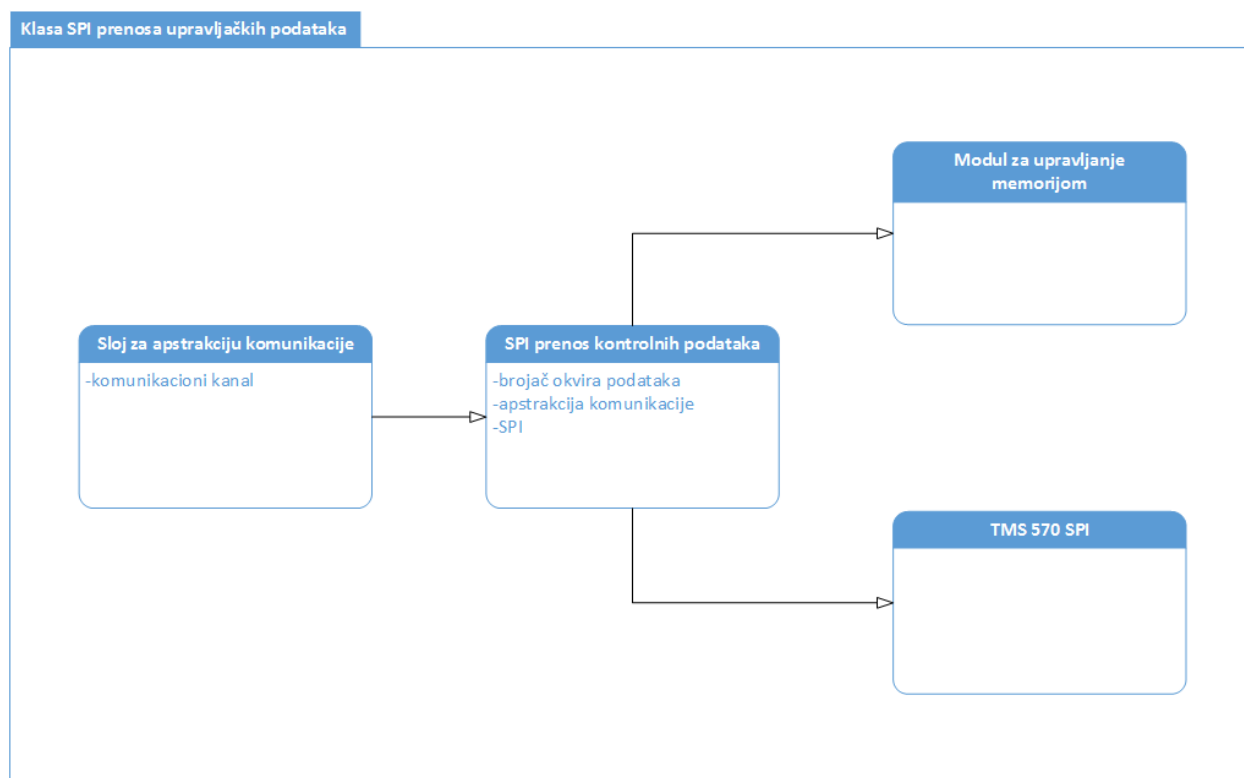
Svaki SPI modul će koristiti svoju SPI magistralu. SPI modul za kontrolne podatke koristi SPI2 magistralu dok će SPI modul za upravljačke podatke koristiti SPI4 magistralu. SPI kontroler prima podatke od sloja za komunikaciju i smešta ih u prijemni bafer pre daljeg prenosa. U suprotnom smeru, podaci se smeštaju u predajni bafer, a zatim se šalju sloju za komunikaciju.

Postoji jedan zadatak SPI kontrolera koji kontroliše prenos na obe SPI magistrale. Kopiranje podataka troši dosta procesorskog vremena, pogotovo kada se podaci kopiraju svakih 0,5ms. Zbog toga je potrebno da se ovo kopiranje odvija u pozadini i da ga vrši modul za pristup memoriji. SPI kontroler samo proverava da li ima paketa spremnih za prenos i ako ima inicira prenos kod modula za pristup memoriji. Vreme potrebno za ovu operaciju ne bi trebalo da prelazi 50us. Prenos podataka preko SPI magistrale je veoma spor, i stoga ne bi smeo da troši mnogo procesorskog vremena.

Oba modula SPI kontrolera rade u okviru istog zadatka čije trajanje ne bi trebalo da prelazi 50 mikro sekundi. Modul za prenos kontrolnih podataka bi trebao da se pokrene prvi kako bi se obezbedila sinhronizacija prenosa.

3.3.1 Modul za prenos upravljačkih podataka

Modul za prenos upravljačkih podataka služi za prenos podataka koji služe za upravljanje sistemom za kontrolu. Na mrežnoj strani podaci se šalju/primaju pomoću kanala sistema za apstrakciju komunikacije. Na strani kontrolnog dela sistema podaci se prenose preko SPI magistrale. Samo poruke sa odgovarajućom oznakom će biti prosleđene na SPI magistralu.



Slika 13. Izgled modula za prenos upravljačkih podataka

Modul za prenos upravljačkih podataka koristi dva kanala modula za upravljanje memorijom za prenos podatak putem SPI magistrale, po jedan za svaki smer prenosa. Takođe koristi četiri kanala sloja za apstrakciju komunikacije 2 za jedan ethernet kanal u jednom pravcu i dva za jedan ethernet kanal u drugom pravcu.

Modul za prenos upravljačkih podataka šalje podatke SPI magistralom u sledećem formatu:

- 4 bajta – zaglavlje
 - 2 bajta - dužina okvira korisnih podataka
 - 1 bajt – broj kanala
 - 1 bajt – tip paketa
- korisni podaci

Navedeni format se koristi u oba smera prenosa podataka, i prilikom slanja i prilikom prijema. Broj kanala je broj fizičkog ethernet kanala za prijem ili slanje podataka. Za tip paketa koristi se vrednost 0x01 za sve ethernet okvire, dok se vrednost 0x02 koristi za sve konfiguracione okvire. Druge vrednosti su rezervisane za buduću upotrebu. Modul za prenos upravljačkih podataka podržava prenos ethernet okvira sa VLAN oznakama. Maksimalna dužina VLAN ethernet okvira je 1518 bajtova, dok se preambula ne prenosi. Moguće je istovremeno slanje samo jednog okvira kontrolnom delu sistema preko SPI magistrale, pa zbog toga prijem sa oba ethernet kanala mora biti zagarantovan.

Modul za prenos upravljačkih podataka treba da implementira sledeće brojače:

- brojač okvira podataka koji se prenose od prvog ethernet kanala do kontrolnog dela sistema
- brojač okvira podataka koji se prenose od kontrolnog dela sistema do prvog ethernet kanala
- brojač okvira podataka koji se prenose od drugog ethernet kanala do kontrolnog dela sistema
- brojač okvira podataka koji se prenose od kontrolnog dela sistema do drugog ethernet kanala
- brojač bajtova koji se prenose od prvog ethernet kanala do kontrolnog dela sistema
- brojač bajtova koji se prenose od kontrolnog dela sistema do prvog ethernet kanala
- brojač bajtova koji se prenose od drugog ethernet kanala do kontrolnog dela sistema
- brojač bajtova koji se prenose od kontrolnog dela sistema do drugog ethernet kanala

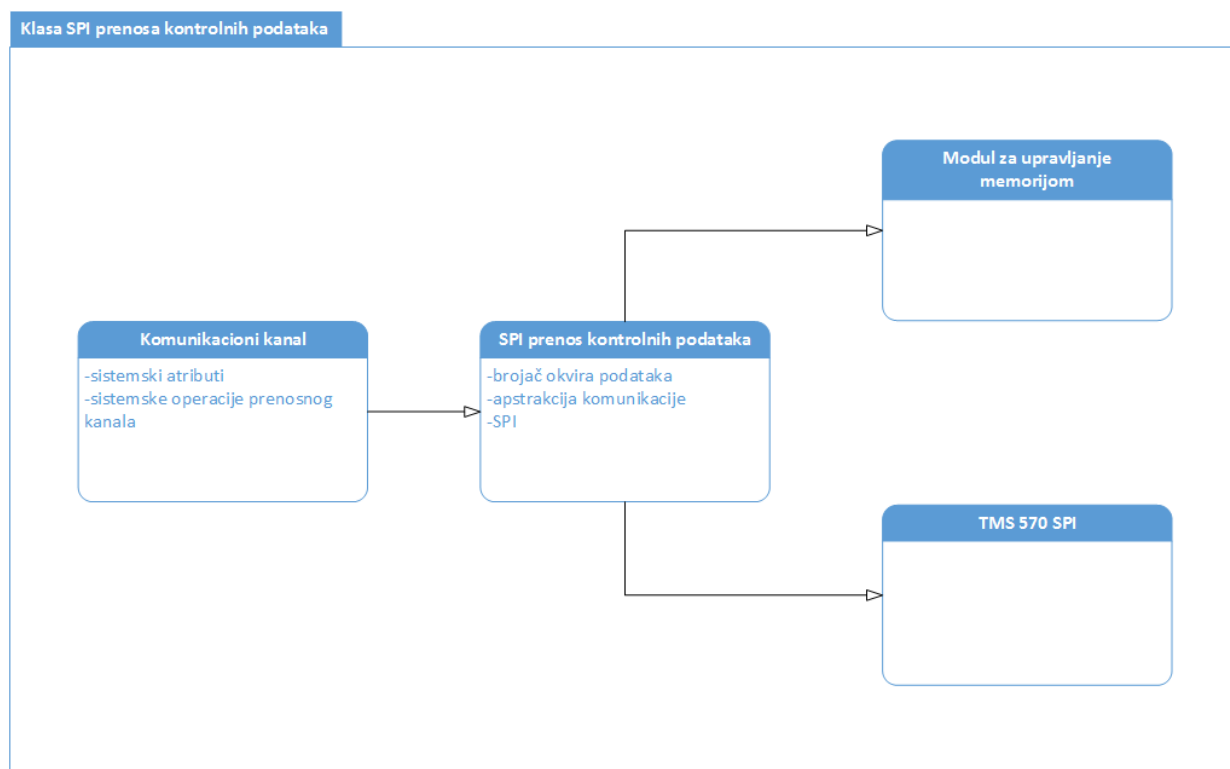
Ukoliko se nikakvi korisni podaci ne šalju magistralom, modul za prenos upravljačkih podataka će slati konfiguracione okvire sledećeg formata:

- 4 bajta – zaglavlje
 - 2 bajta - dužina okvira korisnih podataka = 0x0
 - 1 bajt – broj kanala = 0x00
 - 1 bajt – tip paketa = 0x02 (konfiguracioni okvir)
- korisni podaci

- 1 bajt – oznaka – mora imati vrednost 0x1 za MAC adresu
- 1 bajt – dužina - mora imati vrednost 0x6 za MAC adresu
- 6 bajta – MAC adresa kontrolnog sistema (MAC adresa će prilikom inicijalizacije biti učitana iz odgovarajuće tabele)

3.3.2 Modul za prenos kontrolnih podataka

Modul za prenos kontrolnih podatak putem SPI magistrale služi za prenos podataka koji služe za prenos kontrolnih podataka sakupljenih od strane senzora i aktuatora pomoću kontrolnog sistema.



Slika 14. Izgled modula za prenos kontrolnih podataka

Za prenos podatak SPI magistralom ovaj modul koristi dva kanala modula za upravljanje memorijom, po jedan za oba smera. Takođe koristi 3 kanala modula za apstrakciju komunikacije za prijem i slanje okvira podataka preko mreže, od kojih se jedan koristi za slanje podataka na mrežu, a drugi i treći za prijem podataka sa mreže. Maksimalna količina okvira koji može da prenese modul za prenos kontrolnih podataka je 800 bajta i svaki okvir koji prelazi ovu veličinu će automatski biti odbačen. Takođe je moguće slati samo jedan okvir istovremeno putem ove magistrale.

Modul za prenos kontrolnih podataka omogućava prenos podataka u sledećem formatu:

- 4 bajta – zaglavlje
 - 2 bajta - dužina okvira sa korisnim podacima
 - 2 bajta - broj kanala za prenos
- 800 bajta – korisni podaci (zajedno sa okvirom)

Dužina okvira sa korisnim podacima će se tokom prenosa koristiti za selekciju koji će se okviru prenositi a koji ne, zbog maksimalne veličine prenosnog okvira. Gore navedeni format podataka se koristi za prenos podataka u oba smera.

Potrebno je da modul implementira sledeće brojače:

- brojač okvira koji se prenosi od prvog kanala komunikacionog sloja do modula za kontrolu
- brojač okvira koji se prenosi od modula za kontrolu do komunikacionog sloja
- brojač okvira koji se prenosi od drugog kanala komunikacionog sloja do modula za kontrolu

Uloga ovog brojača je uklanjanje grešaka i integracija u sistem. Korisnik treba da konfiguriše komunikacioni sloj i modul za prenos kontrolnih podataka sledećim parametrima:

- identifikacioni broj kanala za svaki kanal
- tip protokola
- smer toka podataka

Ukoliko korisnik ne želi da koristi neki od kanala moguće ga je definisati na odgovarajući način kako ne bi bio u upotrebi. Modul za prenos kontrolnih podataka kao i modul za prenos upravljačkih podataka treba da poseduju odgovarajuću inicijalizaciju. Takođe svi kanali komunikacionog sloja treba da budu inicijalizovani.

4. Programsko rešenje

U ovom poglavlju su dati detalji implementacije programske podrške za upravljanje SPI magistralom. Biće predstavljen način funkcionisanja modula na nižem nivou, podaci koji su korišćeni u realizaciji kao i opisi realizovanih funkcija.

4.1 Pregled implementacije

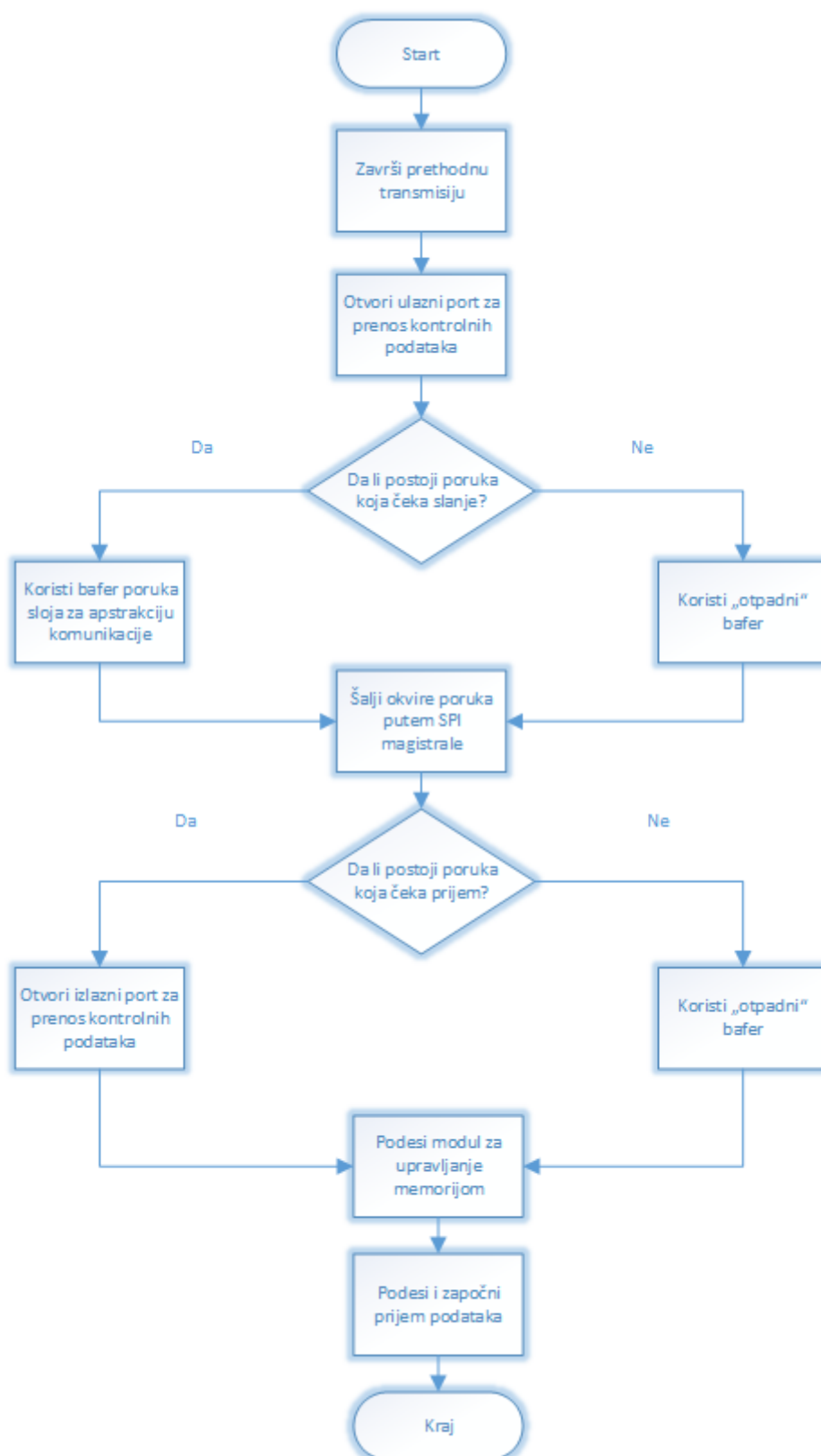
Programska podrška za upravljanje je implementirana kao zadatak koji vrši prenos podataka za kontrolu i upravljanje. Taj zadatak se izvršava periodično i on vrši prenos podataka po SPI magistrali tako što poziva modul za upravljanje memorijom. Ovaj modul takođe koristi modul za apstrakciju komunikacije za pristup okvirima upravljačkih i kontrolnih podataka.

Programska podrška za upravljanje SPI magistralom se sastoji iz 4 dela:

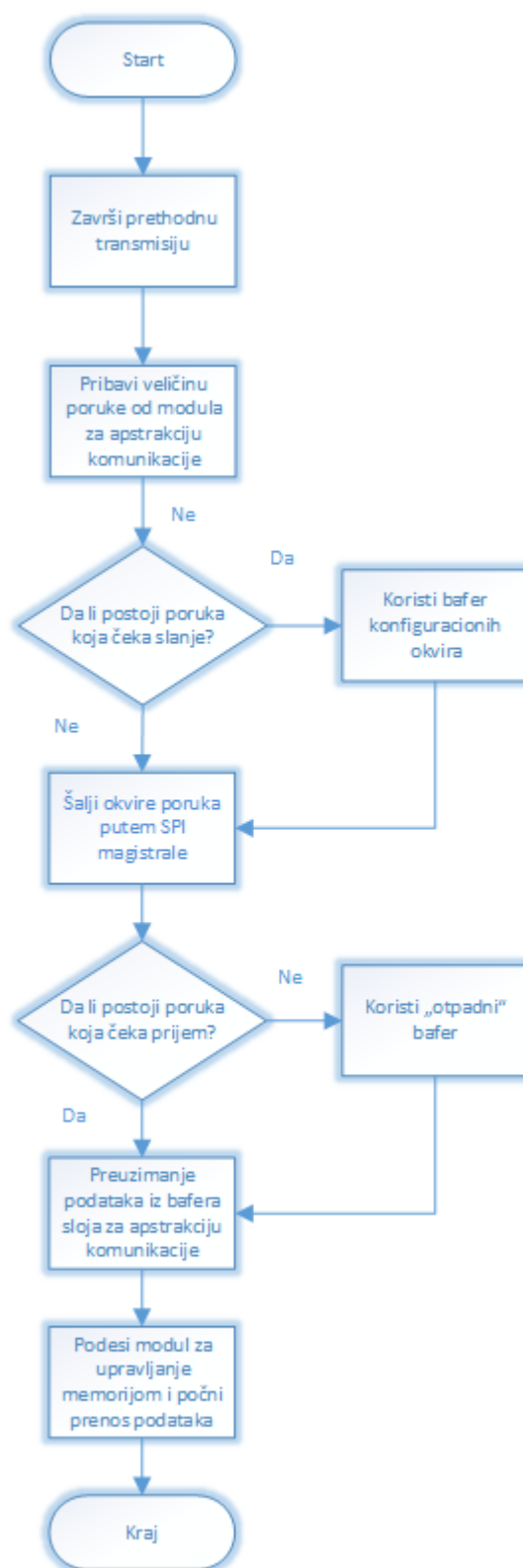
- 1) modul za slanje upravljačkih podataka
- 2) modul za slanje kontrolnih podataka
- 3) inicijalizacija gore navedenih modula
- 4) funkcija za slanje

Postoje dva moda funkcionisanja realizovane programske podrške:

- 1) dupleks mod – prenos podataka u oba smera preko jednog ulaznog i jednog izlaznog virtuelnog ethernet linka
- 2) simpleks mod – samo prenos podataka od mreže preko jednog ulaznog virtuelnog ethernet linka



Slika 15. Aktivnost modula za prenos kontrolnih podataka



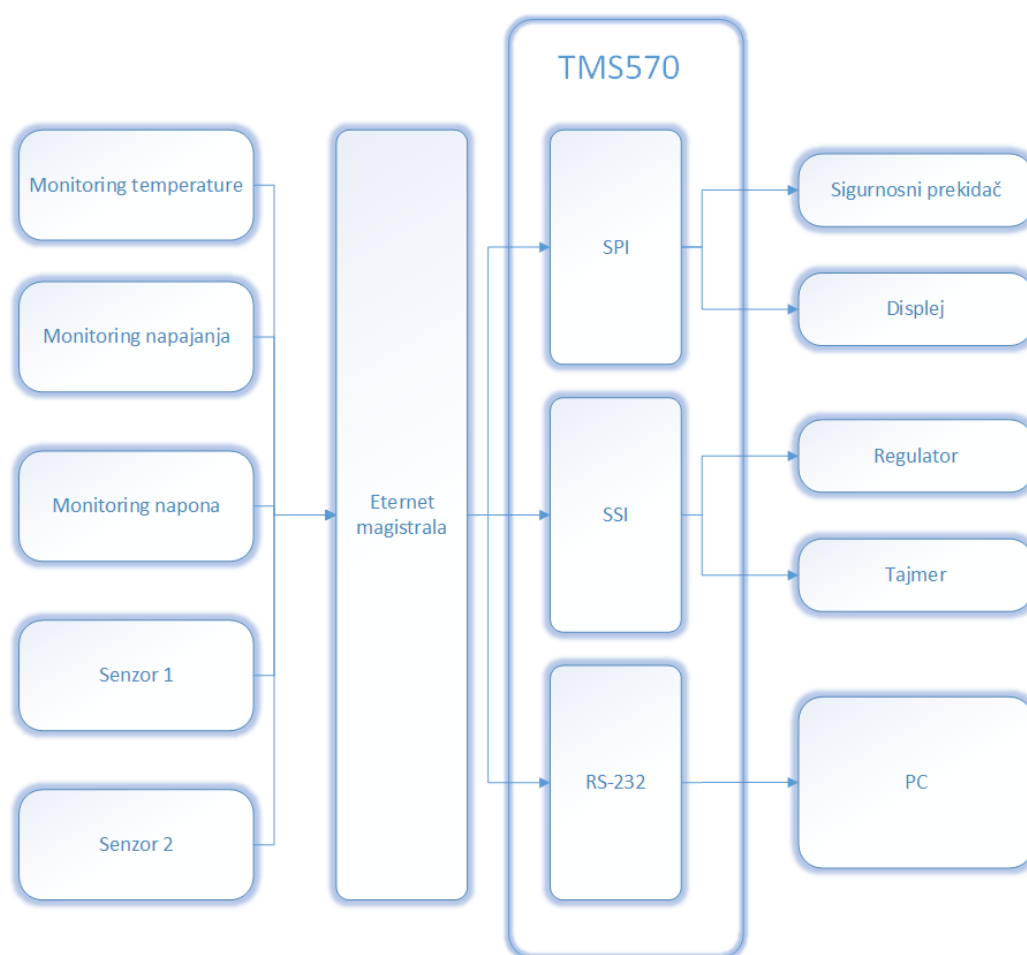
Slika 16. Aktivnost modula za prenos upravljačkih podataka

Kod modula za prenos kontrolnih podataka prvo se vrši provera da li je prethodna transmisija podataka na magistrali završena i ako nije, ona se završava. Zatim se otvara ulazni

port za prenos kontrolnih podataka i proverava se da li trenutno postoji poruka koja čeka na slanje.

Ukoliko ni jedna poruka trenutno ne čeka slanje, onda se otvara poseban bafer iz kog se šalju nasumični beskorisni podaci, jer je potrebno da se na magistrali konstantno odvija saobraćaj. Ako ipak postoji poruka koja trenutno čeka na slanje potrebno je istu preuzeti iz bafera sloja za apstrakciju komunikacije i zatim je poslati SPI magistralom. Posle toga se vrši provera da li neka poruka čeka na prijem, i ako ne čeka ponovo se otvara poseban bafer iz kog se šalju nasumični beskorisni podaci.

Ako postoji poruka koja čeka na prijem, podešava se modul za upravljanje memorijom, konfigurišu se sve neophodne funkcije, i obavlja se transmisija podataka. Kada se ovo okonča, menja se mod prenosa i sistem se vraća u početno stanje.



Slika 17. Primer toka podataka u industrijskom sistemu sa različitim komunikacionim spregama

U slučaju modula za prenos upravljačkih podataka prvi korak je takođe provera da li je prethodna transmisija okončana, a razlika je u tome što u slučaju da transmisija nije završena sistem odmah prelazi u svoje finalno stanje i kreće ciklus iznova.

Ukoliko je prethodna transmisija završena potrebno je preuzeti podržanu veličinu poruke od strane modula za apstrakciju komunikacije. Posle toga se vrši provera da li postoji poruka koja trenutno čeka na slanje, i ukoliko postoji potrebno je pristupiti baferu konfiguracionih okvira u koje će da se smeštaju podaci. Taj korak se preskače ukoliko ni jedna poruka ne čeka slanje i prelazi se na slanje ili okvira sa podacima, ili praznih okvira ukoliko se podaci trenutno ne šalju SPI magistralom.

Zatim se vrši provera da li trenutno postoje poruke koje čekaju na prijem, i ako to nije slučaj koristi se bafer sa beskorisnim podacima koji će održavati magistralu zauzetom. Ukoliko postoje podaci koji čekaju na prijem oni se momentalno preuzimaju iz modula za apstrakciju komunikacije i obavlja se podešavanje modula za upravljanje memorijom posle čega se obavlja transfer podataka. Po završetku prenosa podataka menja se mod rada modula i on prelazi u svoje početno stanje.

Modul za prenos kontrolnih podataka koristi 2 SPI jedinice TMS570 ploče, dok modul za prenos upravljačkih podataka koristi 4 SPI jedinice. Za komunikaciju se takođe koristi 4 kanala modula za upravljanje memorijom.

Programska podrška za upravljanje SPI magistralom prosleđuje ethernet poruke između ethernet mreže i sistema za kontrolu. Mrežnoj strani se pristupa preko modula za apstrakciju komunikacije i kontrolne particije preko SPI magistrale. Čitanje i upis poruka sa modula za apstrakciju komunikacije i SPI magistrale je povereno modulu za upravljanje memorijom. Maksimalna podržana veličina okvira kod modula za prenos kontrolnih podataka je 800 bajta, dok je maksimalna podržana veličina okvira kod modula za prenos upravljačkih podataka je 1518 bajta.

4.2 Strukture podataka korišćenih u realizaciji

CD_SPI_Packet_Header – struktura koja sadrži okvire za kontrolne podatke koji se prenose SPI magistralom i sastoji se od sledećih polja:

<i>Naziv polja</i>	<i>Tip</i>	<i>Opis</i>
length	uint16_t	Dužina korisnih informacija u bajtima
channel	uint8_t	Ethernet broj kanala preko kog se prenose podaci
packet_type	uint8_t	Tip paketa

Tabela 1. Struktura okvira za prenos kontrolnih podataka

MD_SPI_Packet_Header – struktura koja sadrži okvire za upravljačke podatke koji se prenose SPI magistralom i sastoji se od sledećih polja:

<i>Naziv polja</i>	<i>Tip</i>	<i>Opis</i>
length	uint16_t	Dužina korisnih informacija u bajtima
mode	uint16_t	Mod prenosa podataka

Tabela 2. Struktura okvira za prenos upravljačkih podataka

CD_SPI_states – struktura koja sadrži transmisione statistike prenosa upravljačkih podataka:

<i>Naziv polja</i>	<i>Tip</i>	<i>Opis</i>
ch0_in_frm_cnt	uint32_t	Broj okvira upravljačkih podataka primljenih sa mreže na kanalu 0
ch0_out_frm_cnt	uint32_t	Broj okvira upravljačkih podataka poslatih sa mreže na kanalu 0
ch1_in_frm_cnt	uint32_t	Broj okvira upravljačkih podataka primljenih sa mreže na kanalu 1
ch1_out_frm_cnt	uint32_t	Broj okvira upravljačkih podataka poslatih sa mreže na kanalu 1
ch0_in_byte_cnt	uint32_t	Broj bajtova upravljačkih podataka primljenih sa mreže na kanalu 0
ch0_out_byte_cnt	uint32_t	Broj bajtova upravljačkih podataka poslatih na mrežu na kanalu 0

ch1_in_byte_cnt	uint32_t	Broj bajtova upravljačkih podataka primljenih sa mreže na kanalu 1
ch1_out_byte_cnt	uint32_t	Broj bajtova upravljačkih podataka primljenih sa mreže na kanalu 1
ch0_err_cnt	uint32_t	Broj propuštenih okvira na kanalu 0 u slučaju da je došlo do greške u sloju za apstrakciju komunikacije
ch1_err_cnt	uint32_t	Broj propuštenih okvira na kanalu 1 u slučaju da je došlo do greške u sloju za apstrakciju komunikacije

Tabela 3. Struktura sa podacima transmisione statistike prenosa upravljačkih podataka

MD_SPI_states – struktura koja sadrži transmisione statistike prenosa kontrolnih podataka:

<i>Naziv polja</i>	<i>Tip</i>	<i>Opis</i>
in_frame_cnt_0	uint32_t	Broj kontrolnih okvira primljenih na kanalu 0
in_frame_cnt_1	uint32_t	Broj kontrolnih okvira primljenih na kanalu 1
out_frame_cnt_0	uint32_t	Broj kontrolnih okvira poslatih sa kanala 0
out_frame_cnt_1	uint32_t	Broj kontrolnih okvira poslatih sa kanala 1

Tabela 4. Struktura sa podacima o okvirima kontrolnih podataka

4.3 Opis implementiranih funkcija

4.3.1 spi_bridge_init()

Prototip	<i>bool_t spi_bridge_init (void)</i>
Opis	Funkcija za inicijalizaciju sistema. Pozivaju se funkcije za pribavljanje hendlera sloja za apstrakciju komunikacije, a zatim se proverava validnost dobijenih vrednosti. Takođe se poziva funkcija za pribavljanje MAC adrese destinacije.
Parametri	-
Povratna vr.	OK – Operacija je uspešno obavljena FAIL – Operacija nije uspešno obavljena

Tabela 5. Struktura spi_bridge_init funkcije

4.3.2 spi_bridge_task()

Prototip	<i>void spi_bridge_task (void *prm)</i>
Opis	Funkcija koja predstavlja zadatak SPI prenosa podataka. Ona poziva funkcije za prenos kontrolnih i upravljačkih podataka.
Parametri	-
Povratna vr.	void

Tabela 6. Struktura spi_bridge_task funkcije

4.3.3 cd_spi_bridge_get_stats()

Prototip	<i>const volatile cd_spi_stats cd_spi_bridge_get_stats(void)</i>
Opis	Funkcija koja vraća pokazivač na strukturu koja sadrži transisione podatke modula za prenos kontrolnih podataka.
Parametri	-
Povratna vr.	strukturu koja sadrži transisione podatke modula

Tabela 7. Struktura cd_spi_bridge_get_stats funkcije

4.3.4 md_spi_bridge_get_stats()

Prototip	<i>const volatile md_spi_stats md_spi_bridge_get_stats(void)</i>
Opis	Funkcija koja vraća pokazivač na strukturu koja sadrži transmisione podatke modula za prenos upravljačkih podataka.
Parametri	-
Povratna vr.	strukturu koja sadrži transmisione podatke modula

Tabela 8. Struktura md_spi_bridge_get_stats funkcije

4.3.5 spi_send_data()

Prototip	<i>Status_Type spi_send_data (const Spi_Config_Type * Spi_Config_Ptr , Spi_Transmit_Type Transmit , uint8 * Send , uint8 * Receive , uint32 No_Of_Elements)</i>
Opis	Funkcija proverava da li je prethodna transmisija podatak uspešno završena i da li su ulazni parametri validni. Zatim se SPI registar puni podacima sa trenutno aktivnog SPI uređaja. Podaci se odmah zatim na odgovarajući način smeštaju u promenljive posle čega su spremni za prenos.
Parametri	Spi_Config_Ptr – pokazivač na konfiguracionu promenljivu Transmit – prenosna konfiguraciona promenljiva Send – bafer za slanje Receive – bafer za prijem No_Of_Elements – broj elemenata
Povratna vr.	OK – operacija uspešno obavljena FAIL – operacija nije uspešno obavljena

Tabela 9. Struktura spi_send_data funkcije

4.3.6 cd_spi_bridge()

Prototip	cd_spi_bridge (void)
Opis	Ova funkcija predstavlja realizaciju modula za prenos kontrolnih podataka. Funkcija će prvo izvršiti proveru da li je prethodna transmisija podataka na magistrali završena i ako nije, ona će se okončati. Zatim će otvoriti ulazni port i proveriti da li trenutno postoji poruka koja čeka na slanje. Ukoliko ni jedna poruka trenutno ne čeka slanje, onda se otvara poseban bafer iz kog se šalju nasumični beskorisni podaci, (zbog potrebe za konstantnim saobraćajem na magistrali). Poruka koja trenutno čeka na slanje se preuzima iz bafera sloja za apstrakciju komunikacije i zatim se šalje SPI magistralom. Zatim se vrši provera da li neka poruka čeka na prijem, i ako ne čeka ponovo se otvara poseban bafer iz kog se šalju nasumični beskorisni podaci. Ako postoji poruka koja čeka na prijem, podešava se modul za upravljanje memorijom, konfigurišu se sve neophodne funkcije, i obavlja se transmisija podataka. Zatim se menja mod prenosa podataka i sistem se vraća u početno stanje.
Parametri	-
Povratna vr.	-

Tabela 10. Struktura cd_spi_bridge funkcije

4.3.7 md_spi_bridge()

Prototip	md_spi_bridge (void)
Opis	Ova funkcija predstavlja realizaciju modula za prenos upravljačkih podataka. Funkcija na startu proverava da li je prethodna transmisija okončana, i ukoliko nije, izvršavanje funkcije se momentalno završava. Ukoliko je prethodna transmisija završena funkcija zahteva podržanu veličinu poruke od strane modula za apstrakciju komunikacije. Zatim proverava da li postoji poruka koja trenutno čeka na slanje, i ukoliko postoji pristupa baferu konfiguracionih okvira u koje će da se smeštaju podaci. Ako ni jedna poruka ne čeka slanje, prelazi se na slanje ili okvira sa podacima, ili praznih okvira ukoliko se podaci

	trenutno ne šalju SPI magistralom. Funkcija zatim proverava da li trenutno postoje poruke koje čekaju na prijem, i ako to nije slučaj koristi se bafer sa beskorisnim podacima (zbog potrebe za konstantnim saobraćajem na magistrali). Podaci koji čekaju na prijem se momentalno preuzimaju iz modula za apstrakciju komunikacije i obavlja se podešavanje modula za upravljanje memorijom pozivom odgovarajućih funkcija posle čega se obavlja transfer podataka. Posle toga se vrši promena moda rada modula i on prelazi u svoje početno stanje.
Parametri	-
Povratna vr.	-

Tabela 11. Struktura md_spi_bridge funkcije

5. Ispitivanje i verifikacija

U ovom poglavlju opisano je na kojim nivoima i primenom kojih metoda je testiran implementirani modul. Opisano je jedinično testiranje kao najniži nivo testiranja, ali i integraciono testiranje kao i sistemsko testiranje koje predstavlja testiranje najvišeg nivoa. Takođe je dat opis V-modela razvoja softvera kako bi se lakše razumela potreba za datim metodama testiranja.

5.1 Jedinično testiranje SPI kontrolera

Jedinično testiranje (testiranje pojedinačnih modula) predstavlja fazu testiranja neke programske podrške u kojoj se svaka programska komponenta testira nezavisno od ostatka sistema. Ovaj tip testiranja se svodi na proveru funkcionalnosti neke komponente sistema za unapred zadati skup ulaznih podataka koji treba da pokrije sve situacije u kojima komponenta može da se nađe tj. da se simuliraju svi mogući scenariji ponašanja date komponente. Testiranjem se proverava da li se za skup datih ulaznih vrednosti podataka dobijaju očekivani rezultati na izlazu sistema ili se uspešno izvršavaju očekivane akcije. Jedinično testiranje takođe obuhvata proveru korišćenih struktura podataka, logike primenjenih procedura kao i opsega ulaznih i izlaznih podataka [10].

Kao testna platforma za jedinično testiranje razvijanog modula korišćen je ACT – *Automatic Certification Test*. Ova platforma služi za izvršavanje jediničnih testova na računaru uz prikaz izvršavanja testova u komandnom prozoru kao i upis detaljnih informacija u log datoteke. Testovi se mogu izvršavati jedan po jedan sa detaljnim informacijama o izvršenom testu, ali

može se pokrenuti i veća grupa testova, u zavisnosti od potrebe. Takođe, testove je moguće simulirati na računaru, ali po potrebi ih je uz određena podešavanja i opremu moguće izvršiti i na nekoj ciljnoj platformi (u našem slučaju TMS570).

```

variant_cnt += 1
_tc = Test_Case \
( number      = 1
, variant     = variant_cnt
, uut        = "spi_bridge_init"
, description = \
    """
    First three calls to function get_channel_handle are valid.
    """
, setup       = [ g.chanel_hndl ("HDL_INVHDL")
                  , g.chanel_hndl_1 ( a.chanel_struct_1 () )
                  , Function ("get_channel_handle", [ g.chanel_hndl_1 ()
                                                      , g.chanel_hndl_1 ()
                                                      , g.chanel_hndl_1 ()
                                                      , g.chanel_hndl ()
                                                      , g.chanel_hndl ()
                                                      , g.chanel_hndl ()
                                                      , g.chanel_hndl ()
                                                      ])
                  , Function ("routing_table_get_mac_addr", None)
                  ]
, call        = [ Call ("spi_bridge_init", None)
                  ]
, exit        = [ g.be_state ()
                  , a.be_configuration[2] ()
                  ]
, criteria    = [ check_return_value ("spi_bridge_init", "FALSE")
                  , check_for_function_called ("cal_get_channel_handle", 7)
                  , check_for_function_called ("routing_table_get_mac_addr")
                  , check_for_one_parameter ("get_channel_handle", 0, "18", 0)
                  , check_for_one_parameter ("get_channel_handle", 0, "17", 1)
                  , check_for_one_parameter ("get_channel_handle", 0, "19", 2)
                  , check_for_one_parameter ("get_channel_handle", 0, "13", 3)
                  , check_for_one_parameter ("get_channel_handle", 0, "14", 4)
                  , check_for_one_parameter ("get_channel_handle", 0, "15", 5)
                  , check_for_one_parameter ("get_channel_handle", 0, "16", 6)
                  , check_for_function_parameters ("routing_table_get_mac_addr", "[ 13, exit[1] ]")
                  , check_for_variable ("exit[0]", "Be_SpiBridge_State_Error")
                  ]
, verbose     = SHOW_DATA
)

tc.append (_tc)

```

Slika 18. Primer testne skripte za testiranje SPI modula

Test se sastoji iz niza testnih skripti koje se izvršavaju sekvencijalno. Testovi imaju formu pajton skripti koje opisuju sve akcije neophodne za konfigurisanje testa, preuzimanje rezultata kao i proveru pass/fail kriterijuma. U testnoj skripti se definišu funkcije koje poziva testirana funkcija. Njima je moguće zadati vrednost koju će zadata funkcija vratiti kako bi se na taj način uticalo na tok programa. Takođe je moguće menjati i vrednosti promenljivih i struktura (koje nisu lokalne) sa ciljem određivanja toka izvršavanja programa. Tokom izvršenja skripte, moguće je proveriti sledeće parametre:

- povratnu vrednost testirane funkcije

- da li je u toku izvršavanja pozvana neka određena funkcija
- da li u toku izvršavanja nije pozvana neka određena funkcija
- parametre pozivanih funkcija
- vrednost određenih varijabli

Prilikom testiranja programske podrške za upravljanje SPI magistralom kreirano je ukupno 39 testnih varijanti koje su proveravale ispravnost celokupnog modula. 100% pokrivenost koda testovima je verifikovana *Cantata* alatom za proveru pokrivenosti [11].

Sa aspekta nivoa testiranja jediničnim testovima postoje tri vrste testiranja:

- testiranje na niskom nivou – kada se testira samo jedna funkcija
- testiranje na visokom nivou – kada se odjednom testira više funkcija kako bi se testirala interakcija između tih funkcija
- testiranje performansi – merenje najdužeg vremena izvršenja funkcije

<i>Funkcija</i>	<i>Najduže vreme izvršavanja (ms)</i>
spi_bridge_init()	1,35
spi_bridge_task()	1,76
cd_spi_bridge_get_stats()	1,57
md_spi_stats md_spi_bridge_get_stats()	1,26
status_Type _spi_send_data()	1,53
cd_spi_bridge()	0,81
md_spi_bridge()	1,45

Tabela 12. Najduže vreme izvršavanja funkcija

Sa aspekta svrhe testiranja jediničnim testovima postoje dve vrste testiranja:

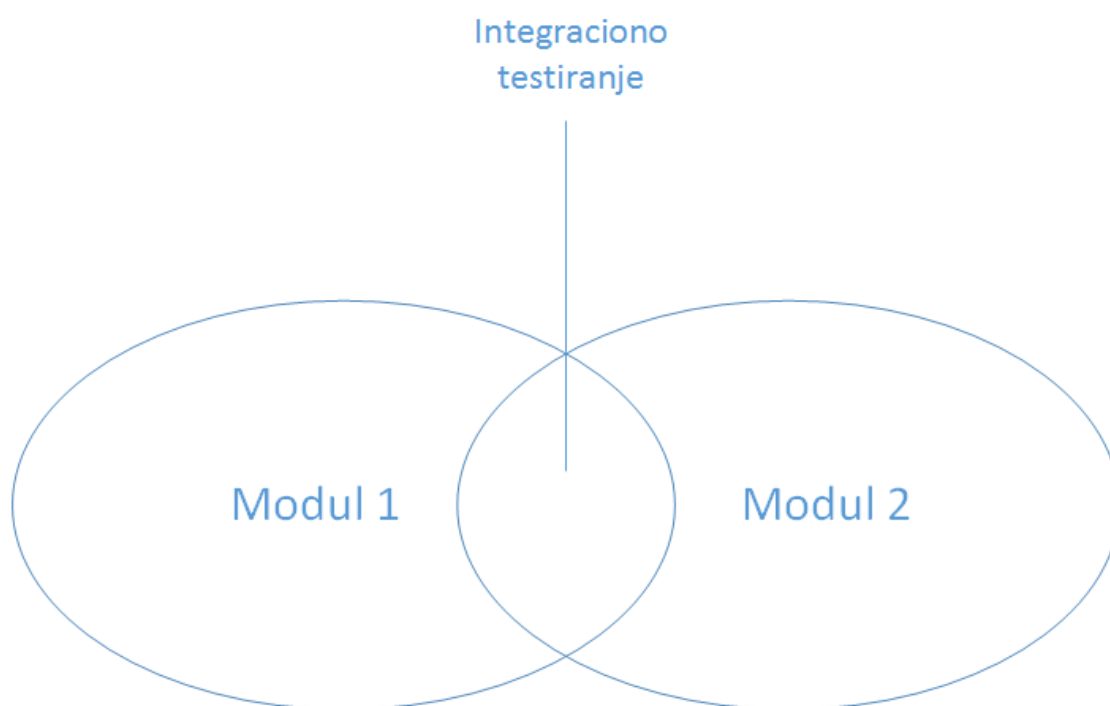
- testiranje robusnosti – testiranje reakcije funkcije na nedozvoljene vrednosti
- regularno testiranje – testiranje ponašanja sa dozvoljenim vrednostima

5.2 Integraciono testiranje SPI kontrolera

Nakon završenog jediničnog testiranja kreće sa povezivanjem modula u jednu celinu i formira se sistem namenjen korisniku. Taj proces se naziva integracija, i ona se vrši dodavanjem komponentata za koje je utvrđeno da ispravno funkcionišu. To znači da se u datom trenutku

različiti moduli nalaze u različitim fazama razvoja, neki se nalaze u fazi kodiranja, neki se nalaze u fazi jediničnog testiranja dok se neki već nalaze u fazi integracije. Zbog toga je vrlo bitno držati se razvojnog plana sistema jer poštovanje redosleda realizacije, testiranja integracije itd utiču na troškove razvoja projekta [12] [13].

Iako je ispravnost individualnih komponenti potvrđena kroz jedinične testove to ne garantuje ispravan rad komponenti kada se one nađu u sprezi sa drugim komponentama. Postoji mogućnost da jedan modul nepovoljno utiče na drugi, da dolazi do gubitka podataka prilikom prenosa, da je došlo do neadekvatne upotrebe tipova podataka ili da greška koja je nastupila u jednom modulu dostigne neprihvatljive vrednosti prilikom prostiranja kroz druge module. Na kraju integracionog testiranja sistema dobili smo celinu koja je spremna za sistemsko testiranje.



Slika 19. Opseg integracionog testiranja

Integraciono testiranje softvera za upravljanje SPI magistralom obuhvatilo je tri vrste provera:

- **Provera podataka izgubljenih kroz spregu** – podrazumeva proveru podataka na izlazu SPI modula i ulazu modula kome se podaci šalju kada je u pitanju slanje podataka, kao i proveru podataka na ulazu u SPI modul i na izlazu modula koji šalje podatke kada je u pitanju prijem podataka. U oba slučaja izvršili smo presretanje podataka na ulazu i na izlazu, a zatim smo ih upisali u datoteku. Tako dobijene datoteke poredimo, i ako je transfer uspešno obavljen one moraju da budu bit-egzaktne. Takođe je potrebno pokriti i slučaj kada je transfer podataka neuspešno izvršen, tj. proveriti ponašanje sistema u ovakvim situacijama. Ovo se testira tako što šaljemo ili primamo blok podataka sa namerno neispravnim okvirom, a zatim očekujemo povratnu poruku o grešci i zahtev za ponovnim slanjem oštećenog bloka.

- **Provera prihvatljive nepreciznosti** – za svaki modul postoji definisana dozvoljena tolerancija greške koju je potrebno proveriti prilikom integracije, s obzirom da greška koja se u granicama normale javila u jednom modulu, može da se iz različitih razloga poveća izvan dozvoljenih granica kada se neki podatak koristi u nekom drugom modulu. U slučaju SPI magistrale, ova provera se svodi na utvrđivanje dozvoljene greške u bitima kod odgovarajućih podataka.
- **Provera globalnih struktura podataka** – može da nastane ukoliko više modula koristi zajedničku memoriju ili ukoliko se podatak na pogrešan način prenosi kroz module. Zbog toga je potrebno proveriti vrednosti deljenih promenljivih u svim modulima koji je koriste. Takođe je potrebno proveriti sve upise u deljenu memoriju u toku jednog ciklusa kako ne bi došlo do prepisivanja nekog relevantnog podatka na nekoj memorijskoj lokaciji.

5.3 Sistemsko testiranje SPI kontrolera

Sistemska testiranje je najviši, završni nivo testiranja. Svrha sistemskog testiranja je provera da li se sistem kao celina ponaša u skladu sa specifikacijom zahteva koja je utvrđena pre početka razvoja sistema. Ovim tipom testiranja se za razliku od funkcionalnih zahteva koji su provereni na nižim nivoima testiranja, proveravaju nefunkcionalni zahtevi kao što su brzina, efikasnost, povezanost sa drugim aplikacijama kao i okruženje u kom će sistem funkcionisati.



Slika 20. Koraci u sistemskom testiranju

Kod sistemskog testiranja ceo razvojni tim je uključen u proces testiranja. Za razliku od nižih nivoa testiranja gde se proverava da li sistem funkcioniše onako kako to želi razvojni tim na ovom nivou se proverava da li su zadovoljeni zahtevi naručioca projekta.

Sistemske testiranje podrazumeva sledeće korake:

- **funkcionalno testiranje** – ovaj korak se izvodi po principu vrlo sličnom „crnoj kutiji“. Funkcionalnim testiranjem se proverava da li sistem obavlja neku funkciju ne obraćajući pažnju na činjenicu koja komponenta u sistemu vrši tu funkciju. Njega vrši tim nezavistan od projekatara i programera koji su realizovali sistem. Ulazi i očekivani izlazi i akcije su unapred poznati i sistem se testira sa kako ispravnim, tako i neispravnim podacima. Prilikom funkcionalnog testiranja ovde realizovanog SPI modula potvrđena je funkcionalnost dela za prenos upravljačkih podataka kroz niz akcija za koje su potrebni upravljački podaci, dok je deo za prenos kontrolnih podataka testiran namernim izazivanjem grešaka u sistemu koje su inicirale promenu stanja na magistrali.
- **testiranje performansi** – se vrši nakon što smo se kroz funkcionalno testiranje uverili da sistem radi prema specificiranim zahtevima korisnika. Postoje brojne podvrste testova performansi, ali u nastavku će biti predstavljene samo one koji su poslužile za testiranje softvera za upravljanje SPI magistralom:
 - testovi konfiguracije – ovim testovima se ispituje ponašanje softvera u različitim hardverskim ili softverskim okruženjima. Kako sistem poseduje čitav spektar konfiguracija za različita okruženja, ovim testovima je neophodno ispitati da li sve konfiguracije zadovoljavaju sistemske uslove.
 - testovi opterećenja – testovi kojima se ocenjuje rad sistema kada se on optereti do svojih operativnih granica u kratkom vremenskom periodu. Ovaj aspekt testiranja je izuzetno bitan za SPI magistralu jer ona uglavnom radi ispod maksimalnog opterećenja ali u određenim trenucima trpi izuzetno veliko opterećenje.
 - vremenski testovi – proveravaju zahteve koji se odnose na vreme izvršavanja funkcija i vreme odziva. Rade se u kombinaciji sa testovima opterećenja i ispituju da li su vremenska ograničenja zadovoljena pri maksimalnom opterećenju.
- **testiranje prihvatljivosti** – izvodi naručilac sistema i njegova svrha je provera da li sistem ispunjava njihove zahteve i očekivanja

- ***instalaciono testiranje*** – predstavlja poslednju fazu sistemskog testiranja i za zadatak ima da pokaže ispravnost sistema na lokaciji i u okruženju u kom će sistem zaista i da radi.

6. Zaključak

U ovom radu opisano je rešenje za prenos podataka preko SPI magistrale u delu industrijskog sistema za nadzor i kontrolu koji se zasniva na Texas Instruments TMS570 hardverskoj platformi. Sistem je realizovan tako da zajedno sa SPI drajverom kontroliše dve instance SPI magistrale. Zbog toga, modul za upravljanje SPI magistralom se sastoji iz dve celine:

- modula za prenos upravljačkih podataka – služi za prenos podataka i instrukcija koje se koriste za upravljanje sistemom
- modula za prenos kontrolnih podataka – služi za prenos podataka koji sadrže informacije o kontrolnim parametrima sistema

Prilikom realizacije ovog softvera, strogo se vodilo računa o performansama i brzini rada sistema, ali i o hardverskim zahtevima modula, pošto se u industrijskim sistemima uglavnom raspolaže skromnim hardverskim resursima. U skladu sa gore pomenutim ograničenjima, takođe je bilo potrebno obratiti pažnju na tipove i strukture podataka, kako zbog što manje potrošnje resursa, tako i zbog pouzdanosti celokupnog sistema.

Ovaj modul je vrlo bitna karika u industrijskom sistemu i s obzirom na činjenicu da je zadužen za prenos podataka koji kontrolišu pouzdanost i pravilno funkcionisanje sistema, pojedinačne module ali i ceo sistem je bilo neophodno detaljno testirati. Ovo je obavljeno kako na nižem nivou, putem jediničnih testova, tako i prilikom integracije modula u sistem kako bi se ispitala kooperacija sa drugim modulima, ali i na najvišem sistemskom nivou kako bi se proverilo ispravno funkcionisanje čitavog sistema.

7. Literatura

- [1] Galloway, B.; Hancke, G.P., "Introduction to Industrial Control Networks," in *Communications Surveys & Tutorials, IEEE* , vol.15, no.2, pp.860-880, 2013
- [2] TMS570LS0x32 16- and 32-Bit RISC Flash Microcontroller Datasheet, Texas Instruments, June 2015
- [3] Vladimir Kovačević, Miroslav Popović: Sistemska programska podrška u realnom vremenu, Univerzitet u Novom Sadu, Fakultet Tehničkih Nauka, 2002
- [4] Leens, F., "An introduction to I²C and SPI protocols," in *Instrumentation & Measurement Magazine, IEEE* , vol.12, no.1, pp.8-13, February 2009
- [5] Jianlong Zhang; Chunyu Wu; Wenjing Zhang; Jiwei Wang, "The design and realization of a comprehensive SPI interface controller," in *Mechanic Automation and Control Engineering (MACE), 2011 Second International Conference on* , vol., no., pp.4529-4532, 15-17 July 2011
- [6] Oruganti, D.N.; Yellampalli, S.S., "Design of power efficient SPI interface," in *Advances in Computing, Communications and Informatics (ICACCI), 2014 International Conference on* , vol., no., pp.2602-2606, 24-27 Sept. 2014
- [7] Using V model for testing, https://insights.sei.cmu.edu/sei_blog/2013/11/using-v-models-for-testing.html, učitano 04.11.2015.
- [8] Deuter, A., "Slicing the V-Model -- Reduced Effort, Higher Flexibility," in *Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on* , vol., no., pp.1-10, 26-29 Aug. 2013
- [9] Liu Shuping; Pang Ling, "The Research of V Model in Testing Embedded Software," in *Computer Science and Information Technology, 2008. ICCSIT '08. International Conference on* , vol., no., pp.463-466, Aug. 29 2008-Sept. 2 2008

-
- [10] Ellims, M.; Bridges, J.; Ince, D.C., "Unit testing in practice," in *Software Reliability Engineering, 2004. ISSRE 2004. 15th International Symposium on* , vol., no., pp.3-13, 2-5 Nov. 2004
 - [11] Cantata, <http://www.qa-systems.com/cantata.html>, učitano 03.10.2015.
 - [12] Karmore, S.P.; Mabajan, A.R., "Universal methodology for embedded system testing," in *Computer Science & Education (ICCSE), 2013 8th International Conference on* , vol., no., pp.567-572, 26-28 April 2013
 - [13] Brar, H.K.; Kaur, P.J., "Differentiating Integration Testing and unit testing," in *Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on* , vol., no., pp.796-798, 11-13 March 2015