Analiza mogućnosti upotrebe mikrokontrolera TMS320F28069 u sustavima upravljanja

Čikeš, Hrvoje

Undergraduate thesis / Završni rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:987185

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

University North Digital Repository

zir.nsk.hr

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI



Sveučilište
Sjever

Završni rad br. 361/EL/2015

Analiza mogućnosti upotrebe mikrokontroleraTMS320F28069 u sustavima upravljanja

Hrvoje Čikeš, 0396/336

Varaždin, srpanj 2016. godine

Sveučilište Sjever

Odjel za elektrotehniku

Završni rad br. 361/EL/2015

Analiza mogućnosti upotrebe mikrokontroleraTMS320F28069 u sustavima upravljanja

Student

Hrvoje Čikeš, 0396/336

Mentor

Miroslav Horvatić, dipl. ing.

Varaždin, srpanj 2016. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL	Odjel za elektrotehniku
PRISTUPNIK	Hrvoje Čikeš
MATIČNI BROJ	0396/336
DATUM	07.09.2015
KOLEGIJ	Automatsko upravljanje
NASLOV RADA	Analiza mogućnosti upotrebe mikrokontrolera TMS320F28069 u sustavima upravljanja
MENTOR	Miroslav Horvatić dipl. ing.
ZVANJE	predavač

ČLANOVI POVJERENSTVA

1. mr.sc. Ivan Šumiga, dipl. ing.
2. Dunja Srpak, dipl. ing.
3. Miroslav Horvatić, dipl.ing.

Zadatak završnog rada

BROJ 361/EL/2015

OPIS

Analizirati arhitekturu mikrokontrolera TMS320F28069. Opisati korišteno razvojno okruženje mikrokontrolera TMS320F28069. Prikazati i analizirati mogućnosti upotrebe mikrokontrolera TMS320F28069 u različitim sustavima upravljanja.

U radu je potrebno:

- opisati arhitekturu mikrokontrolera TMS320F28069,
- opisati rad u korištenom sklopovskom i programskom okruženju mikrokontrolera TMS320F28069,
- kroz programske primjere objasniti mogućnost upotrebe mikrokontrolera TMS320F28069 u različitim sustavima upravljanja,
- prikazati i objasniti sklopovsko povezivanja mikrokontrolera TMS320F28069 u različitim sustavima upravljanja.

ZADATAK URUČEN 15.09.2015

REPUBLIKA HRVATSKA SVEUČILIŠTE SJEVER KOPRIVNICA

POTPIS MENTORA

Predgovor

Zahvaljujem se svima koji su mi svojim zalaganjem, trudom i strpljenjem pomogli u izradi ovoga rada. Naime, ovaj rad nastao je uz profesionalno vodstvo i savjetovanje te požrtvovan rad mentora Miroslava Horvatića, dipl. ing te članova komisije Dunje Srpak, dipl. ing i mr. sc. Ivana Šumige, dipl. ing. Zahvaljujem se svom mentoru na temi za završni rad, na pomoći prilikom odabira literature koju sam koristio za završni rad kao i na strpljenju, korigiranju grešaka, susretljivosti i želji da mi pomogne prilikom izrade završnog rada. Zhvaljujem se svim djelatnicima referade, kao i svim profesorima koji su mi svojom ljubažnoću i radom proteklih godina pomogli u školovanju.

Sažetak

U ovome završnom radu opisane su mogućnosti upotrebe mikrokontrolera TMS320F28069 u različitim sustavima upravljana. Završni rad podijeljen je na dva dijela. U prvom dijelu opisane su svojstva mikrokontrolera, njegova arhitektura te korišteno razvojno okruženje za razvoj programske potpore. Njegova arhitektura opisana je korištenjem blok dijagrama te su ukratko objašnjeni svi njegovi bitni dijelovi. Drugi dio rada sadrži praktične primjere programa koji upravljaju radom mikrokontrolera. Opisani programi služe za dobivanje impulsno širinske modulacije (engl.PWM). Navedena modulacija koristi se za upravljanje energetskim pretvaračima. Prikazani programi omogućavaju namještanje raznih parametara modulacije. Stvarani su i filtrirani visokorezolucijski PWM (Pulse-Width Modulation) signali, objašnjena je analogno digitalnu pretvorba te komparatorska usporedba određenih analognih signala. Za sve navedene programe upravljanja objašnjeni su principi rada te prikazani izmjereni rezultati rada.

Ključne riječi: mikrokontroler TMS320F28069, Code Composer Studio, pulsno širinska modulacija, analogno-digitalna pretvorba, komparator, silazni digitalni pretvarač.

Abstract

In this final thesis is described the ability of using the microcontroller TMS320F28069 in different operating systems. The thesis is divided into two parts. The first part describes the characteristics of the microcontroller, its architecture and used development environment for software development. The architecture is described using block diagrams and briefly explain all its essential parts. The second part contains practical examples of programmes that control the operation of the microcontroller. The programmes described below are used to obtain a pulse width modulation (PWM). Such modulation is used to control power converters. Programmes which have been shown enable adjustment of various parameters modulation and also for creating the filtered high resolution PWM (Pulse-Width Modulation) signals. For all of these control programs, the operating principles, the analog digital conversion and comparator comparison of certain analog signals are explained and the measured results of the work are showed.

Key words: microcontroller TMS320F28069, Code Composer Studio, pulse width modulation, analog-to-digital conversion, a comparator, a downstream digital converter.

Popis korištenih kratica

ADC	Analog-To-Digital Converter Analogno digitalni pretvarač
ALU	Arithmetic Logic Unit Aritmetičko logička jedinica
BOOT ROM	Memorija koja služi za inicijalizaciju sustava
CBC	Cycle-By-Cycle Ciklus po ciklus
CLA	Control Law Accelerator Dodatni procesor u mikrokontroleru zadužen za matematičke operacije
CCS	Code Composer Studio Programsko okruženje tvrtke Texas Instruments namijenjeno za razvoj i ispitivanje programa
CMPA	Counter Compare A Register Usporedbeni A registar
CMPAHR	Compare A Extension Register Usporedbeni A visokorezolucijski ekstenzijski registar
CMPB	Counter Compare B Register Usporedbeni B registar
CPU	Central Processor Unit Centralna procesorska jedinica
CRC	Cyclical Redundancy Check Ciklička provjera podataka
DAC	Digital-To-Analog Converter Digitalno-analogni pretvornik
DBCTL	Dead-Band Control Register Kontrolni registar mrtve zone
DBFED	Dead-Band Falling Edge Count Register Brojački registar padajućeg brida mrtve zone
DBRED	Dead-Band Rising Edge Delay Count Register Brojački registar rastućeg brida mrtve zone
DBCTL[POL SEL]	Dead-Band Polarity Control Register Registar mrtve zone za invertiranje polariteta
DCAEVT	Digital Compare Output A Event Digitalno usporedbeni događaj A
DC-DC	Istosmjerni pretvarač
DCBEVT	Digital Compare Output B Event Digitalno usporedbeni događaj B
DMA	Direct Memory Access Direktan pristup memoriji
ECAN	Enhanced Controller Area Network Modul za povezivanje mikrokontrolera CAN sabirnicom
ECAP	Enhanced Capture Poboljšani modul za precizno mjerenje vanjskih događaja
EEPROM	Electrically-Erasable Programmable Read-Only Memory Električno izbrisiva programibilna ispisna memorija
EOC	End Of Conversion Kraj pretvorbe
EQEP	Enhanced Quadrature Encoder Pulse

	Modul za povezivanje sa inkramentalnim dekoderom
EPWM	Enhanced Pulse Width Modulator
	Poboljšani modul za stvaranje moduliranih signala
FED	Falling Edge Delay
	Kašnjenje padajućeg brida
FFT	Fast Fourier Transform
	Brza Fourierova transformacija
FLASH	Memorija čvrstog diska koja može biti izbrisana i ponovo programirana
FPU	Floating Point Unit
	Jedinica zadužena za matematičke operacije sa decimalnim brojevima
GPIO	General-Purpose Input/Output
	Općenamjenski ulaz/izlaz
I^2C	Inter-Integrated Circuit
	Modul za povezivanje sa uređajima koji imaju Philpsovu I^2C sabirnicu
JTAG	Joint Test Action Group
	Emulacijski kontroler
LED	Light-Emitting Diode
	Svjetlosna dioda
MEP	Micro Edge Positioner
	Tehnologija temeljena mikro rubnom pozicioniranju
McBSP	Multichannel Buffered Serial Port
	Modul za direktno povezivanje s drugim uređajima
MOFSET	Metal Oxide Semiconductor Field Effect Transistor
	Metal oksidni poluvodički tranzistor sa efektom polja
OSHT	One-Shot Trip
	Jedan ciklus Trip
OTP	One Time Programmable
	Memorija dostupna samo za čitanje, ne može se mijenjati
PC	Personal Computer
	Računalo opće namjene
PID	Proportional Integral Derivative
	Proporcionalno integralno derivacijski
PIE	Peripheral Interrupt Expansion Block
	Blok za omogućavanje prekida vanjskih jedinica
PLL	Phase Locked Loop
	Fazno zatvorena petlja
PWM	Pulse-Width Modulation
	Modulirani signal
RAM	Random Access Memory
	Memorija sa nasumičnim pristupom
RED	Rising Edge Delay
	Kašnjenje rastućeg brida
ROM	Rising Edge Delay
	Kašnjenje rastućeg brida
SARAM	Sequential Acess And Random Acess Memory
	Memorija sa sekvencijalnim i nasumičnim pristupom
SCI	Serial Comunication Interface
	Dvožični asinkroni serijski port
SOC	Start-Of-Conversion
	Početak pretvorbe

SPI	Serial Peripheral Interface Visoko brzinski sinkronizirani serijski port
S/W	Software Imediate Start Trenutačni start aktiviran programski
TBCTR	Time-Base Counter Vremensko bazni brojač
TBPRD	Time-Base Period Vremensko bazni period
TZCTL	Trip-Zone Control Register Trip zonski kontrolni registar
TZEINT	Trip-Zone Enable Interrupt Register Trip zonski registar za omogućavanje prekida
TZFLG	Trip-Zone Flag Register Trip zonski registar zastavice
TZSEL	Trip-Zone Select Register Trip zonski selekcijski registar
USB	Universal Serial Bus Univerzalna serijska sabirnica
VCU	Viterbi Complex Unit Jedinica za ubrzanje komunikacije

SADRŽAJ

1. Uvod 1

2. Arhitektura mikrokontrolera TMS320F28069 2

2.1. Sklopovski razvojni sustav F28069 controlSTICK 2

2.2. Objašnjenje arhitekture TMS320F28069 mikrokontrolera 3

2.3. CPU 5

2.4. CLA 5

2.5. Viterbi kompleksna jedinica 5

2.6. FPU 6

2.7. Memorijska sabirnica 6

2.8. Sabirnica vanjskih jedinica 6

2.9. Memorija 7

2.10. PIE Blok 8

2.11. Vanjski prekidi 8

2.12. Unutrašnji zero pin oscilatori i PLL 9

2.13. Određivanje takta rada vanjskih jedinica 9

2.14. Watchdog 9

2.15. GPIO multiplekseri 9

2.16. CPU tajmeri 10

2.17. Kontrolne vanjske jedinice 10

2.18. Komunikacijske vanjske jedinice 11

2.19. Sigurnost 11

3. Korištenje sklopovskog i programskog okruženja mikrokontrolera TMS320F28069 12

3.1. Programsko okruženje CodeComposer Studio 12

3.2. Instaliranje CCS 12

3.3. Početak rada s CCS 12

4. Programski primjeri mogućnosti upotrebe TMS320F28069 mikrokontrolera u različitim sustavima upravljanja 16
4.1. Simetrična pulsno širinska modulacija (PWM) 16
4.1.1. Svrha programa 16
4.1.2. Opis rada programa 17
4.2. Asimetrična pulsno širinska modulacija (PWM) 20
4.2.1. Svrha programa 20
4.2.2. Opis rada programa 21
4.3. Asimetrični PWM sa mrtvim vremenom (DeadBand PWM) 24
4.3.1. Svrha programa 24
4.3.2. Opis rada programa 25
4.4. Okidni PWM 27
4.4.1. Svrha programa 27
4.4.2. Opis rada programa 28
4.5. Filtrirani visokorezolucijski PWM 31
4.5.1. Svrha programa 31
4.5.2. Opis rada programa 32
4.6. Uspoređivanje analognih vrijednosti komparatorom 34
4.6.1. Svrha programa 34
4.6.2. Opis rada programa 35
4.7. Okidni PWM generiran uporedbom analognih signala 37
4.7.1. Svrha programa 37
4.7.2. Opis rada programa 38
4.8. Neprekidna ADC pretvorba 40
4.8.1. Svrha programa 40
4.8.2. Opis rada programa 41
4.9. PWM okidna ADC pretvorba 45

4.9.1. Svrha programa 45
4.9.2. Opis rada programa 46
5. Sklopovsko povezivanje mikrokontrolera TMS320F28069 u različitim sustavima upravljanja 48
5.1. Opis upravljanja istosmjernim pretvaračem korištenjem TMS320F28069M mikrokontrolera 48
5.2. Glavne komponente silaznog digitalnog pretvarača 49
5.3. Upravljački signali mikrokontrolera 49
5.4. Upravljanje mikrokontrolerom 50
6. Zaključak 53
7. Literatura 1
8. Prilozi 2
8.1. Programski kod za simetričnu pulsno širinsku modulaciju [6] 2
8.2. Programski kod za asimetričnu pulsno širinsku modulaciju [6] 4
8.3. Programski kod za asimetrični PWM sa mrtvim vremenom (DeadBand PWM) [6] 5
8.4. Programski kod za okidni PWM [6] 7
8.5. Programski kod za filtrirani visokorezolucijski PWM [6] 9
8.6. Programski kod za uspoređivanje analognih vrijednosti komparatorom [6] 11
8.7. Programski kod za okidni PWM generiran uporedbom analognih signala [6] 14
8.8. Programski kod za neprekidnu ADC pretvorbu [6] 16
8.9. Programski kod za PWM okidnu ADC pretvorbu [6] 19
8.10. Dijagram sustava porodice TMS320F2806x mikrokontrolera [2] 22
8.11. Električna shema silaznog digitalnog pretvarača (Digital Power BoosterPack) [7] 23
9. Popis slika 24
10. Popis tablica 26

1. UVOD

Mikrokontroler je elektronički uređaj koji, slično kao i računalo, ima zadaću da zamijeni čovjeka u upravljanju dijela proizvodnog procesa ili gotovo cijelog proizvodnog procesa. Računala opće namjene kao standardne ulazne jedinice koriste tipkovnicu i miš, a kao standardne izlazne jedinice monitor i pisač. Mikrokontroleri su namijenjeni za upravljanje raznim vrstama procesa, pa se osim sa standardnim ulaznim i izlaznim jedinicama, mogu spojiti i sa velikim brojem nestandardnih ulazno izlaznih sklopova. Razlog tome je što su mikrokontroleri uglavnom dizajnirani za specifične zadaće vrlo raznolike od slučaja do slučaja.

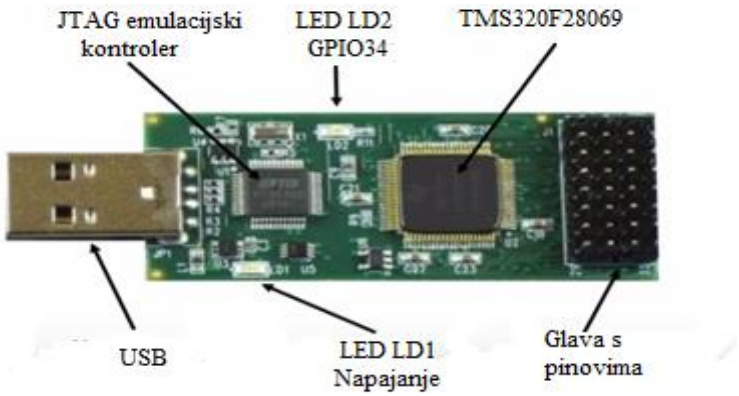
Primjera ima mnogo, od jednostavne regulacije osvjetljenja, alarmnih sustava, pa do upravljanja robotima u industrijskim pogonima. Mikrokontroler je digitalna elektronska naprava u obliku integriranog kruga. Namjena mikrokontrolera je upravljanje uređajima i procesima, pa u sebi ima integriran mikroprocesor, memoriju, digitalne i analogne ulaze i izlaze, digitalne satove (tajmere), brojače, oscilatore, komunikacijske sklopove (interfejse) i druge dodatke poput vanjskih jedinica za koje je nekada bio potreban niz posebnih integriranih krugova. Mikrokontroler normalno radi u kontrolnoj petlji, dakle očitava ulaze i zatim podešava izlaze u skladu sa svojim programom. Petlja se stalno ponavlja dok traje upravljanje procesom.

Mikrokontroler TMS320F28069, prikazan u ovom radu, spada u porodicu C2000 mikrokontrolera Texas Instruments te je optimiziran za široki spektar upravljačkih primjena. On sadrži 32-bitni procesor porodice TMS320C28x, te poboljšane vanjske jedinice što omogućuje dobre performanse za rad u realnom vremenu te mogućnost kontrole i upravljanja elektromotorima, digitalnim napajanjima, pretvaračima u sustavima za dobivanje solarne energije, led osvjetljenjima, pametnom mrežom, sustavima za pozicioniranje radara i mnogim drugim sustavima. Njegove specijalizirane vanjske jedinice koje sadrže PWM (Pulse-Width Modulation) generaciju signala, jedinstvenu ADC (Analog-To-Digital Converter) pretvorbu, poboljšane module za snimanje ECAP (Enhanced Capture), komparatore, itd, ugrađene su u jedinstvenu arhitekturu koja omogućuje bržu, sigurniju i efektivniju kontrolu sustava. U radu je objašnjena osnovna arhitektura mikrokontrolera, a zatim je implementirano i objašnjeno programiranje impulsno-širinskog modulatora, komparatora i modula za analogno-digitalnu pretvorbu.

2. ARHITEKTURA MIKROKONTROLERA TMS320F28069

2.1. Sklopovski razvojni sustav F28069 controlSTICK

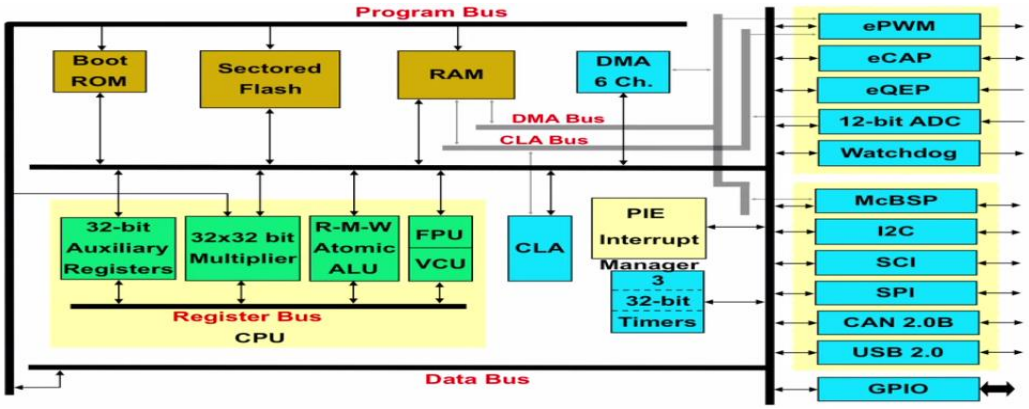
Analiza mogućnosti mikrokontrolera TMS320F28069 vršiti će se korištenjem sklopovskog razvojnog sustava F28069 Piccolo controlSTICK slika 2.1. Navedeni razvojni sustav priključuje se na slobodan USB port osobnog računala, a sadržava JTAG (Joint Test Action Group**)** emulacijski kontroler. USB port sanabdijeva mikrokontroler naponom, a isto tako komunicira sa JTAG emulacijskim kontrolerom. Emulacijski kontroler služi za direktno povezivanje osobnog računala sa TMS320F28069 mikrokontrolerom. JTAG služi za otkrivanje pogrešaka programskog koda, učitavanje programa u TMS320F28069 mikrokontroler te za programiranje FLASH memorije. Za sve te svoje zadaće koristi programski alat kojeg nazivamo CCS (Code Composer Studio), koji će biti detaljnije opisan u trećem poglavlju. Kada se mikrokontroler uključi u USB port svjetleća dioda LED LD1 se uključi te svijetli zelenom bojom što je dokaz da imamo napajanje mikrokontrolera. Također postoji i LED LD2 svjetleća dioda koja je povezana sa pinom GPIO34. Ovisno o funkciji koja joj je pridodana može se uključiti te počinje svijetliti crvenom bojom. Neke od ulazno izlaznih linija mikrokontrolera spojene su na pinove perifernog priključka kojih ima 32. Pinovi perifernog priključka mogu se koristiti kao digitalni/analogni ulazi ili izlazi, ovisno o modulima koji s njima upravljaju. Od 32-pina 14 ih predstavlja višenamjenske ulazi i izlaze GPIO (General Purpose Input Output). Glavni integrirani sklop sklopovskog razvojnog sustava je mikrokontroler TMS320F28069. Negovu arhitekturu i funkcije proučiti će se u sljedećim cjelinama.



Slika 2.1 Sklopovski razvojni sustav F28069 controlSTICK

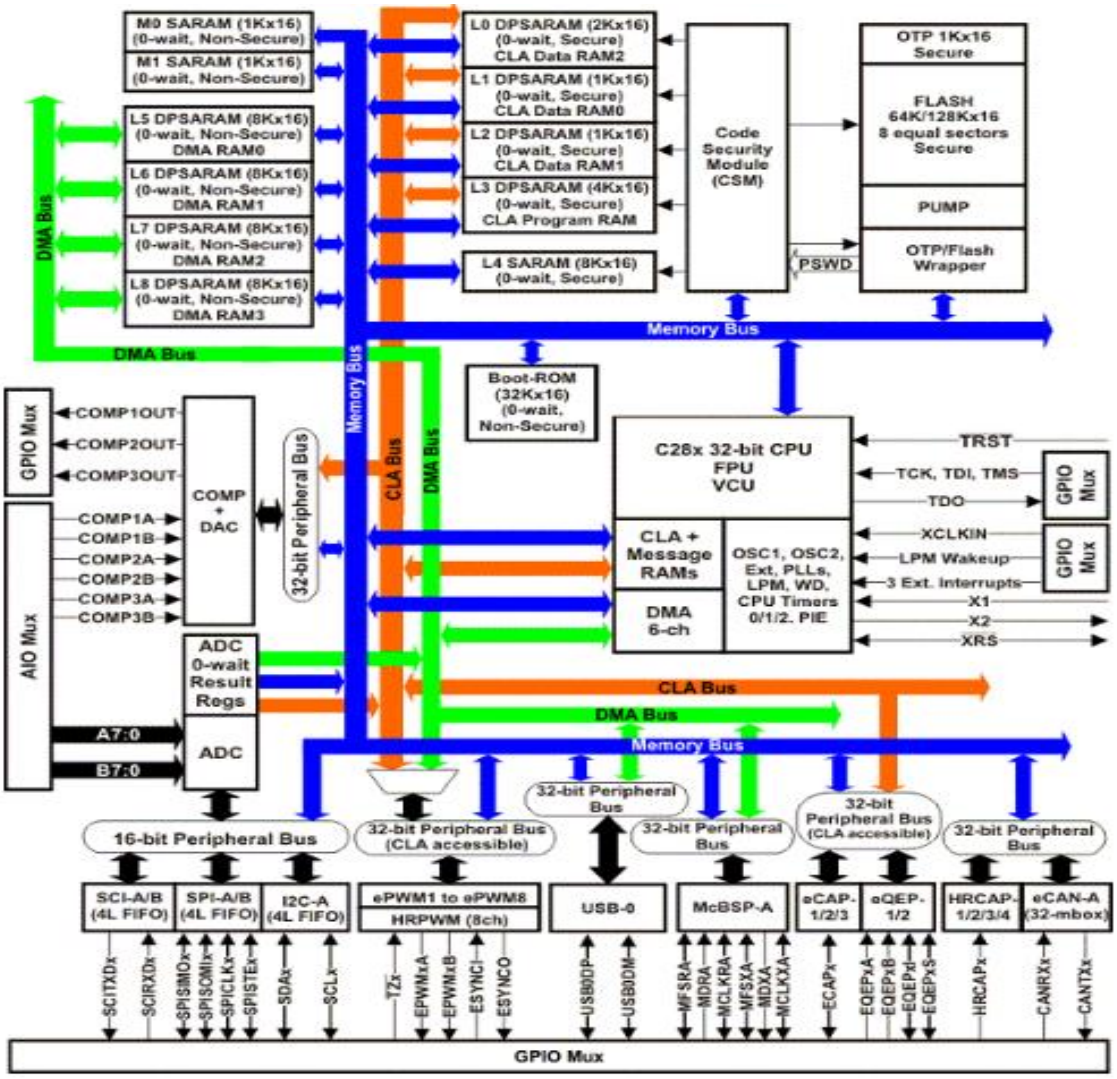
2.2. Objašnjenje arhitekture TMS320F28069 mikrokontrolera

Blok dijagram prikazan na slici 2.2 predstavlja arhitekturu porodice svih TMS320F2806x mikrokontrolera, kojoj pripada korišteni mikrokontroler TMS320F28069. On je dizajniran na principu višesabirničke arhtekture koju također nazivamo i modificirana Harvard arhitektura. [2] Kod takvog tipa arhitekture programska sabirnica odvojena je od podatkovne sabirnice, no postoji veza između njih u donjem lijevom kutu blok dijagrama te skupa čine memorijsku sabirnicu. Ovakav tip arhitekture uvelike povećava performanse mikrokontrolera. U gornjem lijevom području blok dijagrama nalazi se memorijski odjeljak koji se sastoji od BootROM-a (Read-Only Memory), podijeljene FLASH memorije i RAM memorije. Također se može primjetiti da se uz njih nalazi šest kanalni DMA (Direct Memory Access) koji ima svoju vlastitu sabirnicu. U donjem lijevom dijelu može se vidjeti izvršna sekcija CPU (Central Processor Unit), koja se sastoji od 32x32 bitnog harvardskog množila, atomske aritmetičke logičke logičke jedinice ALU, FPU (Floating Point Unit) jedinice za računanje decimalnih brojeva te VCU (Viterbi Complex Math Unit) jedinice koja ubrzava kompleksne algoritme i komunikaciju do 7 puta. CLA (Control Law Accelerator) odnosno dodatni procesor je nezavisna i odvojena jedinica koja ima svoje posebne sabirnice. Vanjske jedinice smještene su na desnoj strani blok dijagrama. Gornji set predstavlja vanjske kontrolne jedinice te se sastoji od: EPWM (Enhanced Pulse Width Modulator) , ECAP (Enhanced Capture), EQEP (Enhanced Quadrature Encoder Pulse), 12-bitnog ADC (Analog-To-Digital Converter) i Watchdog tajmera. Donji set predstavlja komunikacijske vanjske jedinice i sastoji se od: McBSP (Multichannel Buffered Serial Port), I2C (Inter-Integrated Circuit), SCI (Serial Comunication Interface), SPI (Serial Peripheral Interface), ECAN (Enhanced Controller Area Network), USB2.0 (Universal Serial Bus) te na kraju GPIO. PIE (Peripheral Interrupt Expansion Block) omogućava prekide vanjskih jedinica. Također CPU (Central Processor Unit) ima tri 32-bitna tajmera opće namjene. [5]



Slika 2.2 Pojednostavljeni blok dijagram TMS320F28069 mikrokontrolera [5]

Istu strukturu prikazanog pojednostavljenog blok dijagrama može se prikazati malo kompleksnijim funkcijskim blok dijagramom prikazanim na slici 2.3.



Slika 2.3 Funkcijski blok dijagram TMS320F28069 mikrokontrolera [2]

U nastavku će biti detaljnije opisani sljedeći osnovni dijelovi mikrokontrolera.

2.3. CPU

Mikrokontroler TMS320F28069 ima 32-bitnu centralnu procesorsku jedinicu vrlo dobrih performansi koja se može učinkovito programirati u programskom jeziku C/C++. On omogućava korisnicima da razvijaju ne samo upravljačke programe u jezicima visoke razine, nego i da učinkovito implementiraju matematičke algoritme koristeći programski jezik C/C++. Centralna procesorska jedinica je vrlo učinkovita ne samo u izvršavanju upravljačkih zadaća, već i u matematičkim proračunima. Ta učinkovitost otklanja potrebu za drugim procesorom u mnogim sustavima. Njegova 32-bitna jezgra radi na 90 Mhz te ima mogućnost procesiranja čak i visoko rezolucijskih numeričkih problema. S obzirom da ima brz odziv na prekide te da automatski sprema sadržaj kritičnih registara, može obrađivati i nesinhronizirane događaje sa minimalnim kašnjenjem. [2]

2.4. CLA

CLA sklop mikrokontrolera je nezavisni 32-bitni matematički akcelerator napravljen za operacije s decimalnim brojevima, te radi paralelno s glavnim procesorom CPU. On ima posebnu sabirničku i memorijsku strukturu te fetch mehanizam. Fetch mehanizam je logika koja mu omogućuje čitanje podataka unaprijed i privremeno skladištenje u memoriji da bi se poboljšale performanse. Moguće je zadati 8 pojedinačnih CLA zadataka. Svaki zadatak započinje programskim pozivom ili zahtjevom vanjskih jedinica kao što su: ADC, EPWM, ECAP, EQEP ili CPU tajmer 0. CLA izvršava prvi zadatak po prioritetu dok ne završi. Kad završi obradu zadatka, glavni procesor CPU je obaviješten sa prekidnim događajem PIE te CLA automatski započinje sa sljedećim zadatkom koji je po prioritetu prvi. CLA također može i direktno pristupiti ADC, EPWM, HRPWM, EQEP i EQEP registrima. Upotrebom CLA broj ukupnih ciklusa potrebnih za izvršavanje programa je smanjen, što rezultira bržim kontrolnim petljama te ostavlja 80% vremena centralnog procesora za izvršavanje ostalih zadataka. [2]

2.5. Viterbi kompleksna jedinica

VCU (Viterbi Complex Unit) jedinica specijalno je dizajnirana da ubrza performanse u telekomunikacijskim aplikacijama. Arhitektura mikrokontrolera TMS320F28069 je već optimizirana za matematičku obradu i rad u realnom vremenu, no sa VCU također su poboljšane i komunikacije. Zahvaljujući tome može se kontrolirati jedan ili više motora u realnom vremenu, a istovremeno komunicirati i s drugim

uređajima. Brza Fourierova transformacija FFT (Fast Fourier Transform) koristi se u širokom spektru telekomunikacija, kao i u algoritmima za obradu signala. 16-bitni FFT za određenu obradu trebati će 20 ciklusa procesora, no skupa sa VCU-om trebati će samo 5 ciklusa. Digitalni filteri također se koriste da povećaju pouzdanost podataka . VCU može izračunati kompleksno množenje koeficijenata u jednom ciklusu, dok bi normalno trebalo 10 ciklusa. CRC (Cyclical Redundancy Check) vrše provjeru cjelovitosti podataka nad velikim podatkovnim blokovima. Provjera bloka podataka veličine 10 bajtova bez korištenja VCU trajala bi 250 ciklusa, dok sa VCU traje samo 10 ciklusa. Uglavnom VCU je složena matematička jedinica koja ubrzava složene kompleksne komunikacijske algoritme i do sedam puta.

2.6. FPU

FPU (Floating Point Unit) jedinica dio je centralnog procesora specijalno dizajnirana da odradi matematičke operacije s decimalnim brojevima. Tipične njene operacije su: zbrajanje, oduzimanje, množenje, dijeljenje, korjenovanje te pomicanje bitova.

2.7. Memorijska sabirnica

Kao i kod dosta drugih mikrokontrolera, višestruke sabirnice se koriste kako bi mogle razmjenjivati podatke između memorije, vanjskih jedinica te CPU-a. Memorijska sabirnica sadrži programsku sabirnicu za čitanje (read bus), podatkovnu sabirnicu za čitanje (data read bus) te podatkovnu sabirnicu za upis (data write bus). Podatkovna sabirnica za čitanje i upis sadrži 32 adresne linije i 32 podatkovne linije. Višestruka sabirnička arhitektura (Harvard bus arhitecture) omogućuje čitanje i upis podataka u jednom ciklusu. Sve vanjske jedinice i memorije koje su priključene na memorijsku sabirnicu moraju se držati prioriteta memorijskog pristupa. [2]

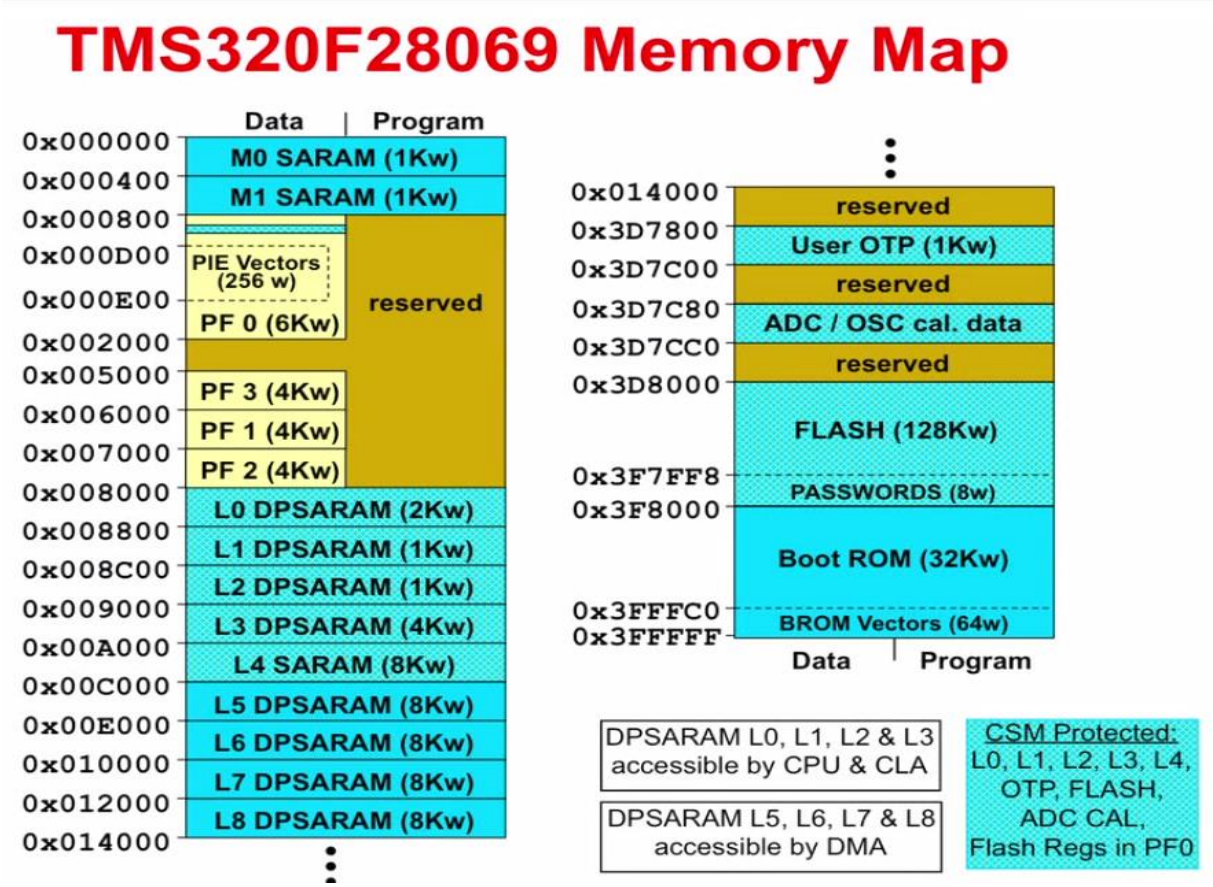
2.8. Sabirnica vanjskih jedinica

Kako bi se omogućilo povezivanje različitih vanjskih jedinica na mikrokontroler TMS320F28069, rezvijen je standard za povezivanje vanjskih jedinica. Sabirnice različitih vanjskih jedinica multipleksiraju se na memorijsku sabirnicu. Ona sadrži 16 adresnih linija, 32 podatkovne linije te povezane kontrolne signale. [2]

2.9. Memorija

Mikrokontroler TMS320F28069 koristi susjedne memorijske mape što je također poznato kao i vezana Neumann-ova arhitektura. Ovakav tip memorijske mape pogodan je za programiranje višim programskim jezicima.

Memorijska arhitektura mikromontrolera TMS320F28069 prikazana je na slici 2.4. Granica između podataka i programa smještena je na vrhu memorijske mape sa proširenim memorijskim blokovima na podatkovni i programski prostor. Na vrhu memorijske mape imamo dva bloka RAM-a 1K x 16 koji su nazvani M0 i M1. Kako su oni pohranjeni i u podatkovnom i u programskom prostoru, korisnik ih može koristiti za izvršenje programskog koda ili za pohranu varijabli. Ispod njh se vide blokovi vanjskih jedinica PF0-PF3 (Peripheral Frames). To je područje gdje se nalaze registri vanjskih jedinica kao i PIE blok koji služi za omogućavanje prekida. Sljedeće područje je područje L memorije koje sadrži 48K x 16 jednopristupnog RAM-a. L0, L1 i L2 blokove koristi CLA i CPU iz podatkovnog prostora, dok L3 blok koristi iz programskog prostora. Blokovi L5, L6, L7 i L8 imaju mogućnost direktnog pristupa memoriji te se mogu koristiti za programski prostor. Sljedeći memorijski blok nazvan je OTP (One Time Programmable) te koristi prostor od samo 1KB za kalibraciju ADC-a i oscilatora. FLASH memorija ili FLASH EEPROM (Electrically-Erasable Programmable Read-Only Memory) je vrsta EEPROM memorije. Promatrani mikrokontroler sadrži 128K x 16 ugrađene FLASH memorije podijeljene u osam 16K x 16 sektora. Korisnik može pojedinačno izbrisati program i potvrditi FLASH sektor, ne dirajući ostale sektore. Međutim, nije moguće koristiti jedan sektor za raliziranje FLASH algoritama koji brišu ili programiraju druge sektore. BOOT ROM memorija smještena je na dnu memorijske mape te služi za inicijalizaciju rada sustava. [5]



Slika 2.4 Prikaz memorijske mape [5]

2.10. PIE Blok

PIE blok služi za multipleksiranje brojnih prekidnih događaja u manji set prekidnih ulaza te može podržati 96 prekida. Prekidi su grupirani u blokove od 8 te je svaka grupa povezana sa 12 CPU-ovih prekidnih linija (INT1-INT12). Svaki od 96 prekida podržan je sa njegovim vlastitim vektorom namjenskog RAM-a kojeg korisnik može prebrisati. Prioritet prekida kontroliran je sklopovski i programski. Svaki prekid se može pojedinačno omogućiti ili onemogućiti unutar PIE bloka. [2]

2.11. Vanjski prekidi

Mikrokontroler TMS320F28069 podržava tri vanjska prekida (XINT1-XINT3). Svaki od prekida može se namjestiti za negativni, pozitivni, pozitivni i negativni brid koji služe za okidanje te također mogu biti omogućeni ili onemogućeni. Ovi prekidi sadrže 16-bitne uzlazne brojače, koji se resetiraju na nulu kada je odgovarajući prekidni brid detektiran. Ne postoje posebno određeni pinovi za vanjske prekide, već oni mogu prihvatiti ulaze od GPIO0-GPIO31 pinova. [2]

2.12. Unutrašnji zero pin oscilatori i PLL

Frekvencija rada mikrokontrolera može se namjestiti preko dva unutrašnja Zero-pin oscilatora ili preko kristala pričvršćenog na strujni krug oscilatora. Kako bi se mogla namjestiti frekvencija rada procesora koristi se PLL sklop. PLL je sistem zatvorene petlje u kojem je unutarnji oscilator kontroliran na način da zadržava fazu i vrijeme vanjskog periodičnog signala koristeći povratnu vezu. Takav način rada omogućava dobivanje 16 različitih omjera brzina takta rada procesora mikrokontrolera. PLL omjeri se mogu mijenjati i tokom rada programa te omogućuju korisniku smanjivanje frekvencije rada procesora ako je dovoljna manja brzina rada. [2]

2.13. Određivanje takta rada vanjskih jedinica

Unutarnji satovi svake vanjske jedinice se mogu pojedinačno uključiti ili isključiti da bi se smanjila potrošnja struje ako vanjska jedinica nije u upotrebi. Također dodatno se može skalirati sistemski sat u odnosu na CPU sat. [2]

2.14. Watchdog

Watchdog tajmer je elektronički tajmer koji se koristi da detektira i oporavi mikrokokontroler od određenih poremećaja i nepravilnosti u radu. TMS320F28069 mikrokontroler ima dva watchdoga. Prvi je CPU-Watchdog koji prati jezgru procesora, a drugi NMI-Watchdog koji prati sistemski sat. Korisnički softver mora resetirati brojač CPU-Watchdoga u određenom vremenu inače će se generirati reset procesora. CPU-Watchdog se također može u potpunosti i isključiti ako je to potrebno. NMI-Watchdog se uključuje samo u slučaju greške sistemskog sata te može generirati prekid ili reset. [2]

2.15. GPIO multiplekseri

Velika većina signala vanjskih jedinica je multipleksirano sa GPIO (General-Purpose Input/Output) općenamjenskim ulazno-izlaznim signalima. Ovo omogućuje korisniku korištenje pina kao GPIO-a, ako signali vanjskih jedinica ili određenih funkcija nisu korišteni. Nakon reseta GPIO pinovi su definirani kao ulazi. Korisnik može pojedinačno programirati svaki pin za GPIO mod ili kao signal vanjskih jedinica.

2.16. CPU tajmeri

CPU-tajmeri 0,1 i 2 su identični 32-bitni tajmeri sa podesivim periodima i 16-bitnim preskaliranim satom. 32-bitni tajmeri rade u silaznom modu te generiraju prekid kad brojač dođe na nulu. Brojač se smanjuje kad se brzina CPU sata podijeli sa preskaliranom vrijednošću. Kad brojač dođe na nulu, on se automatski nadopuni sa 32-bitnom vrijednošću perioda. CPU-tajmer 0 je namijenjen za općenitu upotrebu te je povezan s PIE blokom. CPU-tajmer 1 je također za općenitu upotrebu i može biti spojen na INT13 CPU-a. CPU-timer 2 je rezerva za sistemski bios te je spojen na INT14 CPU-a. Ako ne koristimo bios sustava, tada se CPU-timer 2 može koristiti za unutarnje oscilatore te vanjske satove. [2]

2.17. Kontrolne vanjske jedinice

Kontrolne vanjske jedinice sadrže EPWM, ECAP, EQEP, ADC, komparator i HRCAP.

- EPWM odnosno poboljšana PWM vanjska jedinica podržava nezavisno/komplementarnu generaciju PWM signala, podešavajuću mrtvu zonu na uzlaznim i padajućim bridovima, okidni mehanizam baziran na usporedbi komparatorom te prekidni mehanizam,
- ECAP (Enhanced Capture) odnosno poboljšano hvatanje je neophodno u sustavima gdje je potrebno precizno mjerenje vanjskih događaja. Koristi se za mjerenje brzine rotirajućih strojeva, vremena između pulseva senzora, za detektiranje struje ili napona koje daju određeni strujni ili naponski senzori,
- EQEP (Enhanced Quadratur Eencoder Pulse) modul koristi se za direktno povezivanje sa linearnim ili rotirajućim inkrementalnim enkoderom da bi dobili poziciju, smjer te informaciju o brzini rotacijskog stroja,
- ADC modul je 12-bitni pretvarač. On ima 11 nesimetričnih kanala koji su izvedeni na pinove mikrokontrolera, te također ima dvije jedinice za uzrokovanje i zadržavanje da bi se postiglo istovremeno uzrokovanje,
- Svaki komparatorski modul sastoji se od analognog komparatora i unutarnje 10-bitne reference napona koja može poslužiti umjesto jednog od dva moguća analogna ulaza.

2.18. Komunikacijske vanjske jedinice

Mikrokontroler TMS320F28069 podržava sljedeće tipove serijske komunikacije: SPI, SCI, I^2C, ECAN, McBSP i USB.

- SPI (Serial Peripheral Interface) je visoko brzinski sinkronizirani serijski port koji se koristi za kratke udaljenosti. Obično se koristi za komunikaciju između mikrokontrolera i neke druge vanjske jedinice ili drugog procesora,
- SCI (Serial Comunication Interface) je dvožični asinkroni serijski port koji se obično naziva UART (Universal Asynchronous Receiver/Transmitter) te je u pogledu razmjene podataka dosta sličan sa SPI,
- I^2C (Inter-Integrated Circuit) vanjska jedinica omogućuje povezivanje između mikrokontrolera i drugih uređaja koji su usklađeni sa Philipsovom I^2C sabirnicom. Obično se koristi za povezivanje nisko brzinskih vanjskih jedinica sa mikrokontrolerom,
- ECAN (Enhanced Controller Area Network) vanjska jedinica je napravljena da bi se omogućila komunikacija između mikrokontrolera bez posredstva osobnog računala,
- McSB (Multichannel Buffered Serial Port) omogućava direktno povezivanje sa drugim DSP-ima Texas Instrumentsa kao i drugim uređajima u sustavu. Njegovi registri za odašiljanje i slanje prilagođeni su direktnom pristupu memorije što uvelike smanjuje posao za ovu jedinicu. Najviše se koristi u audio obradi signala,
- USB

2.19. Sigurnost

Mikrokontroler TMS320F28069 podržava visoke razine sigurnosti, kako bi se korisnički softver i programi što bolje zaštitili. Da bi onemogućili neovlaštenom osoblju pristup sve je zaštićeno 128-bitnom lozinkom. Na taj način onemogućuje se pristup pregleda memorije preko JTAG porta, izvršavanje programskog koda iz vanjske memorije ili dodavanje neželjenog softvera koji bi promijenio sigurnosne memorijske sadržaje. [2]

3. KORIŠTENJE SKLOPOVSKOG I PROGRAMSKOG OKRUŽENJA MIKROKONTROLERA TMS320F28069

3.1. Programsko okruženje CodeComposer Studio

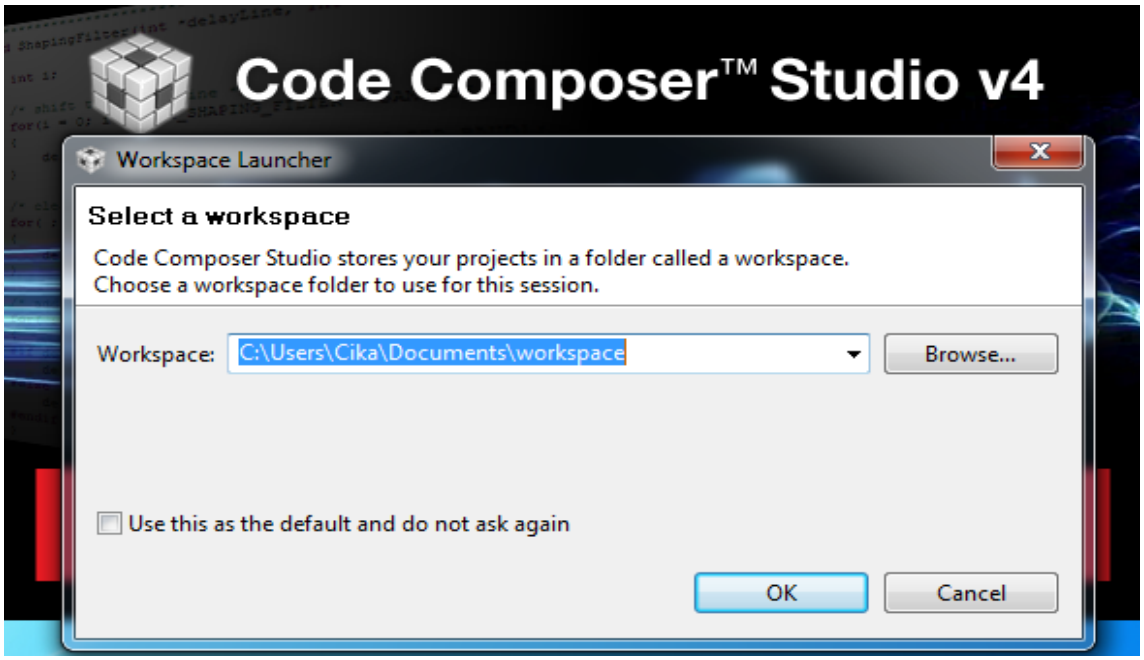
CCS (Code Composer Studio) je integrirano razvojno okruženje koje podržava Texas Instruments mikrokontrolere. CCS sadrži mnogostruke alate koji se mogu koristiti za razvoj i otkrivanje pogrešaka pri stvaranju aplikacija. CCS sadrži optimizirani C/C++ kompajler, editor izvornog koda, okruženje za izradu projekta i otkrivanje greški te još mnogo drugih mogućnosti. Integrirano korisničko sučelje olakšava razvoj programskog koda.

3.2. Instaliranje CCS

Minimalna platforma za instaliranje CCS-a je osobno računalo s 1.5GHz, 1GB RAM-a te 300 MB slobodnog prostora na tvrdom disku. Nakon što se u računalo stavi instalacijski DVD, potrebno je odabrati ikonu setup_CCS_4.2.1.exe kako bi započela instalacija. Tijekom instalacije otvarati će se određeni komunikacijski prozori na koje je potrebno odgovoriti. Nakon što se instalacija izvrši pojaviti će se ikona CCS verzije 4.2.1 preko koje se može pokrenuti program. Valja napomeniti da će nakon instaliranja proizvod biti licenciran samo 30 dana. Da bi se dobila trajna licenca, potrebno se registirati na službenim stranicama Texas Instrumentsa te zatražiti licencu. Nakon što se odobri trajna licenca, bit će poslana na korisnikov mail.

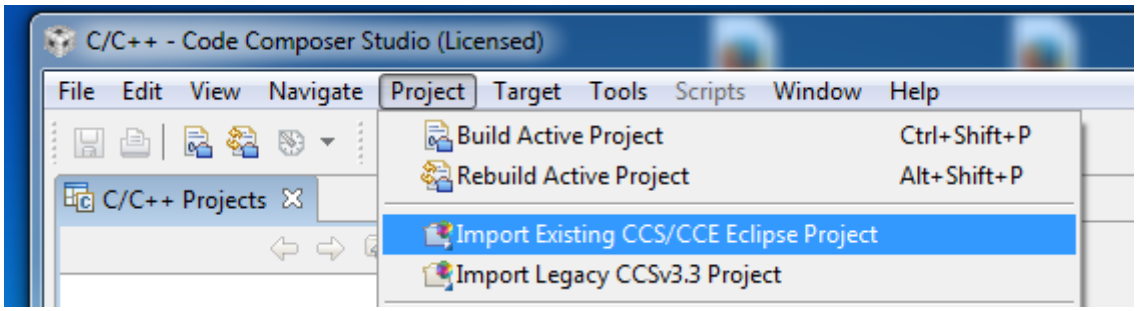
3.3. Početak rada s CCS

Nakon prvog pokretanja CCS pojaviti će se prozor dobrodošlice nazvan Welcome Page. Na toj stranici vidi se nekoliko linkova koji mogu biti otvoreni kako bi dali više informacija o korištenju CCS-a. Preporučeno je odvojiti neko vrijeme za proučavanje tih informacija, a osobito dijela Getting Started kako bi se što više što više olakšalo korištenje. Nakon što se zatvori Welcome Page otvoriti će se novi prozor koji će ponuditi odabir mape sa radnim područjem workspace, kao što je prikazano na slici 3.1.



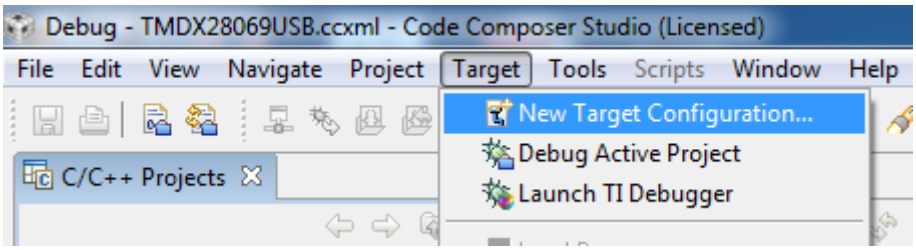
Slika 3.1 Odabir radnog područja zvanog Workspace

Workspace je područje gdje se spremaju korisničke postavke CCS-a. To uključuje sve postavke projekta što je posebno dobro ako se istovremeno radi na nekoliko projekata, jer će se na taj način memorijski i grafički prozori za promatranje varijabli sačuvati te biti dostupni prilikom ponovnog otvaranja CCS-a. Prilikom gašenja CCS-a sve postavke projekta automatski se snimaju u workspace. Nakon što se odabere workspace započinje se sa otvaranjem prvog projekta. To se radi klikom miša na project te nakon toga odabirom Import Existing CCS/CCE Eclipse Project kao što je prikazano na slici 3.2.



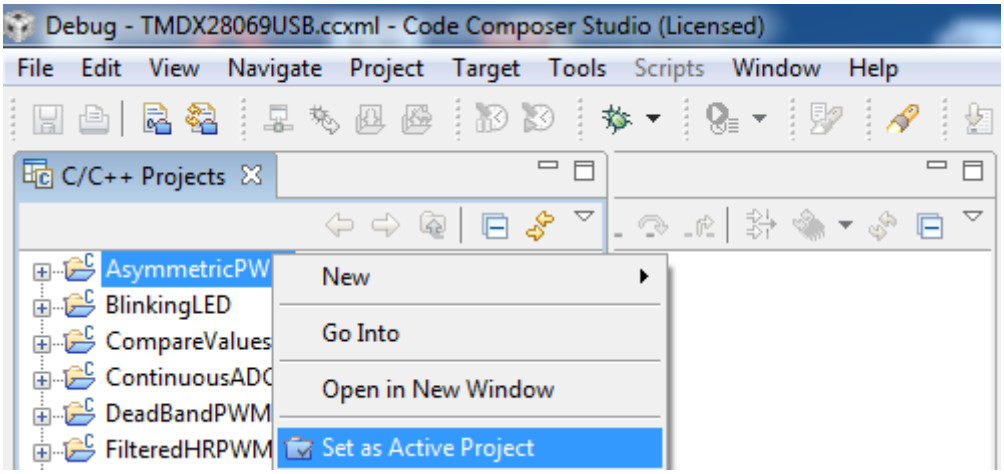
Slika 3.2 Otvaranje postojećih projekata

Na ovaj način otvaraju se postojeći projekti koji se nalaze na računalu, te se prenose u CCS u C/C++ perspektivi. Sljedeći korak je povezivanje mikrokontrolera sa CCS-om. To se radi klikom na target te zatim na New Target Configuration kako što je prikazano na slici 3.3.



Slika 3.3 Povezivanje CCS-a sa mikrokontrolerom

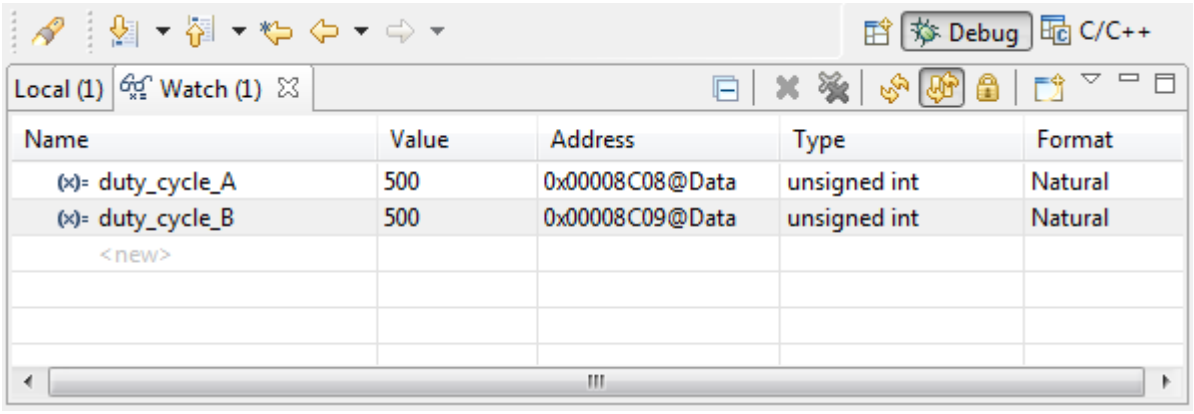
Nakon toga otvoriti će se izbornik gdje je potrebno odabrati verziju razvojnog sklopovlja, kao i tip emulacijskog čipa. Sljedeći korak je povezivanje New Target Configuratin sa određenim projektima koji se pokreću preko mikrokontrolera. Kako je mikrokontroler sad povezan sa CCS, a projekti učitani, kreće se sa njihovom izradom. U ovom radu postoji više projekata koji se izvršavaju, međutim ne može se izvršavati više njih istovremeno. Zbog toga se klikne mišom na određeni projekt i postavlja kao aktivnog (Set as Active Project). Za izradu aktivnog projekta potrebno je stisniti na zelenu tipku koja je u obliku kukca, a zove se debug. Prikaz postavljanja aktivnog projekta i tipke debug prikazan je na slici 3.4.



Slika 3.4 Aktiviranje projekta

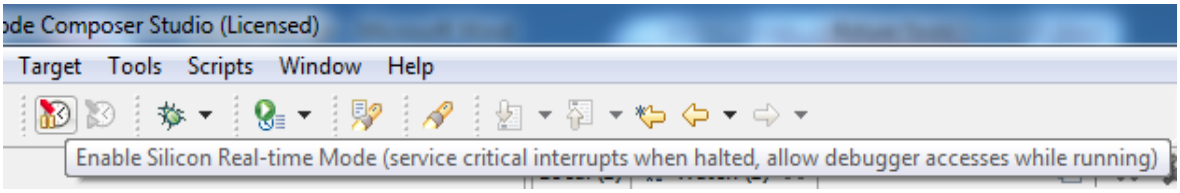
Klikom na tipku debug odabrani projekt automatski se povezuje te učitava programski kod na priključeno razvojno sklopovlje. Valja napomeniti da se programski kod ne izvršava u ovom trenutku nego je u fazi čekanja. Ovisno od projekta do projekta određene varijable se mogu unijeti u prozor za gledanje varijabli kojeg nazivamo Watch Window, a prikazan je na slici 3.5. Na taj način mogu se mijenjati vrijednosti tih varijabli bez potrebe za promatranjem glavnog dijela programskog koda. Jedini uvjet je da CCS bude u debug perspektivi, jer u C/C++ perspektivi nije omogućen pristup prozoru za promatranje varijabli već samo glavnom programskom kodu. Prozor za gledanje varijabli

prikazan je slikom 3.5. U gornjem desnom rubu prozora vidi se kako je moguće prijeći iz Debug u C/C++ perspektivu.



Slika 3.5 Prikaz prozora za promatranje i mijenjanje varijabli

Za promatranje promjene varijabli u realnom vremenu potrebno je stisnuti na ikonu koja je u obliku sata što je prikazano slikom 3.6.



Slika 3.6 Uključenje moda za rad u realnom vremenu

Sad je napravljeno sve što je potrebno da bi se krenilo sa izvršavanjem projekta, odnosno programskog koda koji je učitan u mikrokontroler. Početak izvršavanja se radi tako da da se pritisne tipka play koja je zelene boje. Do nje se nalazi tipka za pauzu, kao i crvena tipku koja služi da se programski kod makne iz pripravnog stanja za izvršenje.



Slika 3.7 Prikaz tipki za upravljanje i uklanjanje projekta

4. PROGRAMSKI PRIMJERI MOGUĆNOSTI UPOTREBE TMS320F28069 MIKROKONTROLERA U RAZLIČITIM SUSTAVIMA UPRAVLJANJA

4.1. Simetrična pulsno širinska modulacija (PWM)

4.1.1. Svrha programa

Ovaj program konfigurira EPWM2 vanjsku jedinicu za generiranje par simetričnih PWM signala. Ta dva PWM signala mogu se nezavisno modulirati te su konfigurirani kao dva simetrična signala koja rade u uzlazno-silaznom brojačkom modu. Ovaj par PWM signala se šalje na izlaze preko EPWM2A i EPWM2B pinova. [6]

Tablica 4.1 Prikaz sklopovskog povezivanja mikrontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

Nema spoja	Osciloskop

Postavka softvera radi se tako da nakon što se u CCS unese program i učita u pločicu, u prozor za promatranje varijabli (watch window) dodaju se sljedeće varijable:

- duty_cycle_A (ova varijabla namiješta širinu impulsa PWM2A kanala)
- duty_cycle_ B (ova varijabla namiješta širinu impulsa PWM2B kanala)

Obje varijable početno su postavljene na vrijednost 250 od ukupnog perioda 500 što ujedno i znači da će PWM signali biti pola perioda uključeni, a pola ne.

4.1.2. Opis rada programa

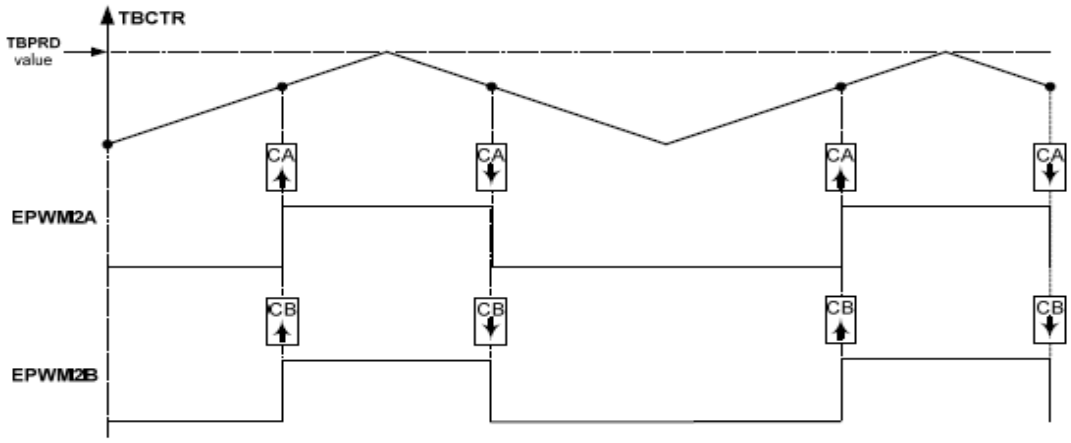
Ovim programom EPWM vanjska jedinica konfigurirana je da radi u uzlazno-silaznom brojačkom modu. U uzlazno-silaznom modu vremensko-bazni brojač TBCTR (Time-Base Counter) koji je drektno vezan sa frekvencijom takta procesora, broji od nule te se povećava dok nije dostignuta vrijednost perioda TBPRD (Time Base Period). Kad je vrijednost perioda dostignuta, vremensko-bazni brojač smanjuje svoju vrijednost dok ne dođe do nule. U trenutku kad je došao na nulu ponovo će ponoviti isti postupak. Oblik generiranog pulsno širinskog valnog oblika određen je načinom na koje su određene akcije akcijskog kvalifikacijskog modula (Action Qualifier Module) konfigurirane te tako utječu na određene događaje brojača. Programski kod za ovu konfiguraciju prikazan je slikom 4.1, a prikaz načina stvaranja PWM simetričnih signala slikom 4.2.

Za ovaj projekt uzeti su sljedeći parovi akcija-događaj:

- Postavi PWM2A na događaj TBCTR= CMPA, uzlazno brojanje (CAU)
- Obriši PWM2A na događaj TBCTR=CMPA, silazno brojanje (CAD)
- Postavi PWM2B na događaj TBCTR= CMPB, uzlazno brojanje (CBU)
- Obriši PWM2B na događaj TBCTR= CMPB, silazno brojanje (CBD) [6]

```
// Akcije postavljanja
EPwm2Regs.AQCTLA.bit.CAU = AQ_SET; // Postavi PWM2A na događaj A,
uzlazno brojanje
EPwm2Regs.AQCTLA.bit.CAD = AQ_CLEAR; // Obriši PWM2A na događaj A,
silazno brojanje
EPwm2Regs.AQCTLB.bit.CBU = AQ_SET; // Postavi PWM2B na događajB,
uzlazno brojanje
EPwm2Regs.AQCTLB.bit.CBD = AQ_CLEAR; // Obriši PWM2B na događaj B,
silazno brojanje
```

Slika 4.1 Programski kod akcijskog kvalifikacijskog modula za dva simetrična PWM-a [6]



Slika 4.2 Princip stvaranja simetričnih PWM signala [6]

Frekvencija PWM signala za uzlazno-silazni mod brojača računa se po sljedećim izrazima:

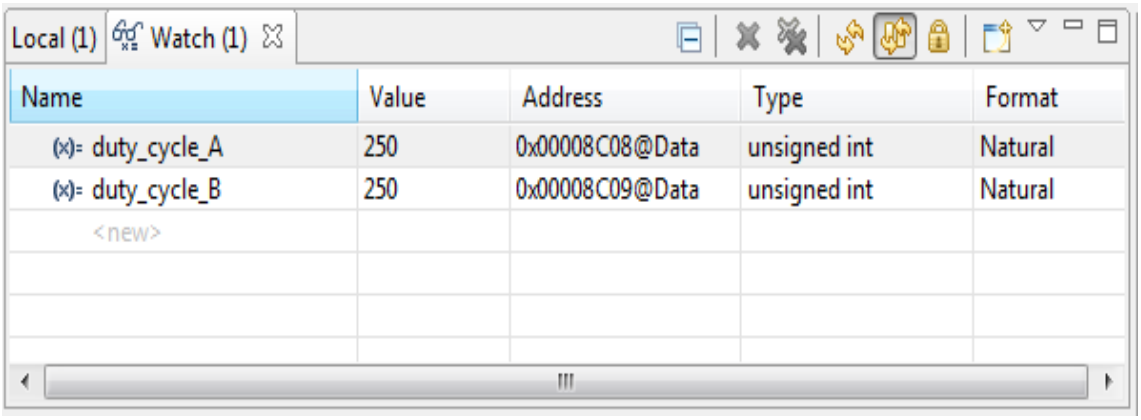
$$T_{PWM} = (2 * TBPRB) * T_{TBCLK} \quad (4\text{-}1)$$

$$F_{PWM} = 1/(T_{PWM)}) \quad (4.2)$$

Gdje je TBPRD period postavljen početno u programu na vrijednost 500, a T_{TBCLK} je period takta procesora koji ima vrijednost 1/80 MHz. Na osnovu izraza 4-1 i 4-2 generirana PWM frekvencija (F_{PWM}) za početni primjer ima vrijednost 80 kHz. Stvoreni PWM simetrični signali promatraju se preko osciloskopa spojenog na EPWM2A i EPWM2B pinove. Oblik PWM signala mijenja se promjenom koda akcijskog kvalifikacijskog modula, dok se frekvencija mijenja promjenom perioda u programu. Širina impulsa za svaki PWM kanal može se neovisno mijenjati iz prozora za promatranje varijabli mijenjanjem vrijednosti duty_cycleA i duty_cycleB, kao što je prikazano na slici 4.4.

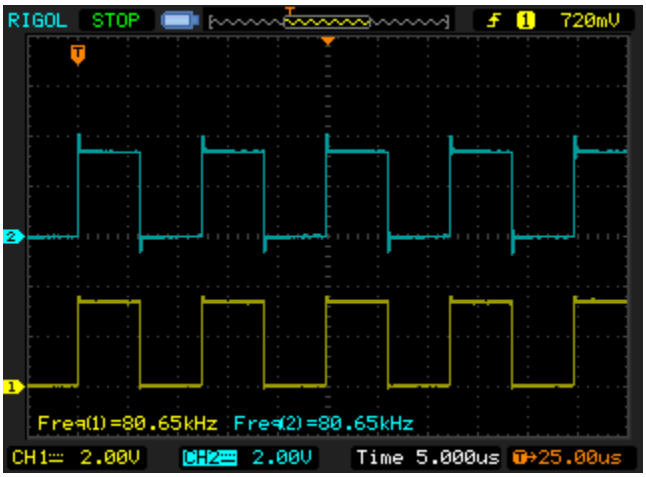
```
for(;;)
{
EPwm2Regs.CMPA.half.CMPA = duty_cycle_A; // Dodaj duty_cycle_A
u watch window
EPwm2Regs.CMPB = duty_cycle_B; // Dodaj duty_cycle_B u watch
window
```

Slika 4.3 Programski kod za dodavanje varijabli u watch window [6]



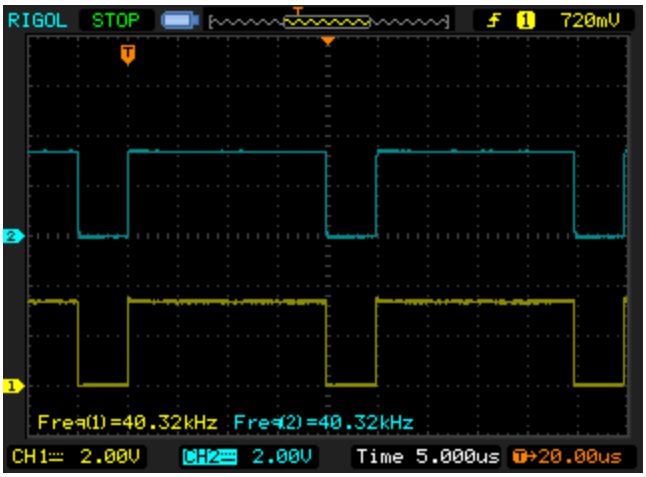
Slika 4.4 Varijable prikazane u watch window

Prilikom snimanja na osciloskop vrijednost perioda će biti 500, pa će frekvencija moduliranih valova iznositi 80 kHz. Vrijednost za namještanje širine impulsa postavljena je na 250 te će modulirani signali biti pola perioda uključeni, a pola isključeni slika 4.5.



Slika 4.5 Simetrični PWM signali frekvencije 80 kHz

U sljedećem mjerenju, slika 4.6 vrijednost TBPRD perioda postavljena je na 1000, dok će se vrijednost za namještanje širine impulsa postaviti na 200 (duty_cycle_A i duty_cycle_B).



Slika 4.6 Simetrični PWM signali frekvencije 40 kHz

Vremensko-bazni brojač koji radi u uzlaznom-silaznom modu najprije počinje brojiti od nule te kad je dostigao zadanu vrijednost 200 akcijski kvalifikacijski modul postavlja akciju, odnosno uključuje oba signala. Zatim brojač nastavlja u uzlaznom modu brojiti do 1000 što je zadana vrijednost TBPRD perioda, U trenutku kad je dostigao TBPRD period, počinje raditi u silaznom modu smanjivajući svoju vrijednost. Kad je dosegao vrijednost 200 u silaznom modu on ponovo postavlja akciju, ali sada isljučuje oba signala. Nakon toga brojač nastavlja brojiti do nule te kad je dostigao nulu ponovo počinje raditi u uzlaznom modu ponavljajući iznova cijeli postupak. Iz navedenog se može vidjeti da će uključenost signala trajati 1600 dok će isključenost trajati 400. Na temelju toga zaključuje se da će signali biti 80% vremena jednog perioda uključeni, a 20% vremena perioda isključeni. Također je vrijednost TBPRD perioda povećana na 1000, odnosno udvostručena u odnosu na prethodno mjerenje. Frekvencija će se zbog toga prepoloviti te će iznositi 40 kHz.

4.2. Asimetrična pulsno širinska modulacija (PWM)

4.2.1. Svrha programa

Ovaj program konfigurira EPWM2 vanjsku jedinicu za generiranje par komplementarnih odnosno asimetričnih PWM signala. Ti PWM signali se mogu nezavisno modulirati i raditi će za razliku od simetričnih valnih oblika, u uzlaznom brojačkom modu brojača. Za izlaze se koriste pinovi EPWM2B i EPWM2A prikazani u tablici 4.2. [6]

Tablica 4.2 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

Nema spoja	Osciloskop

Što se tiče softverske postavke, nakon što se učita program u pločicu kao i u prethodnom primjeru za simetričnu modulaciju, u prozor za prikaz varijabli unose se sljedeće varijable:

- duty_cycle_A (Ova varijabla namiješta širinu impulsa EPWM2A kanala).
- duty_cycle_ B (Ova varijabla namiješta širinu impulsa EPWM2B kanala).

Početno su u programu obje varijable namještene na vrijednost 500 od ukupnnog TBPRD perioda 1000.

4.2.2. Opis rada programa

U ovome programu PWM vanjska jedininca konfigurirana je na način da radi u uzlaznom brojačkom modu. U uzlaznom brojačkom modu PWM brojač TBCTR, koji je direktno povezan sa frekvencijom takta procesora, broji od nule do postavljene vrijednosti perioda TBPRD. Kada brojač dostigne vrijednost perioda TBPRD, postavlja se ponovo na nulu te počinje brojati ponovo. Oblik generiranog PWM valnog oblika određen je akcijama akcijskog kvalifikacijskog modula na način da te akcije imaju određeno ponašanje na određene događaje. Za ovaj program korišteni su sljedeći događaji:

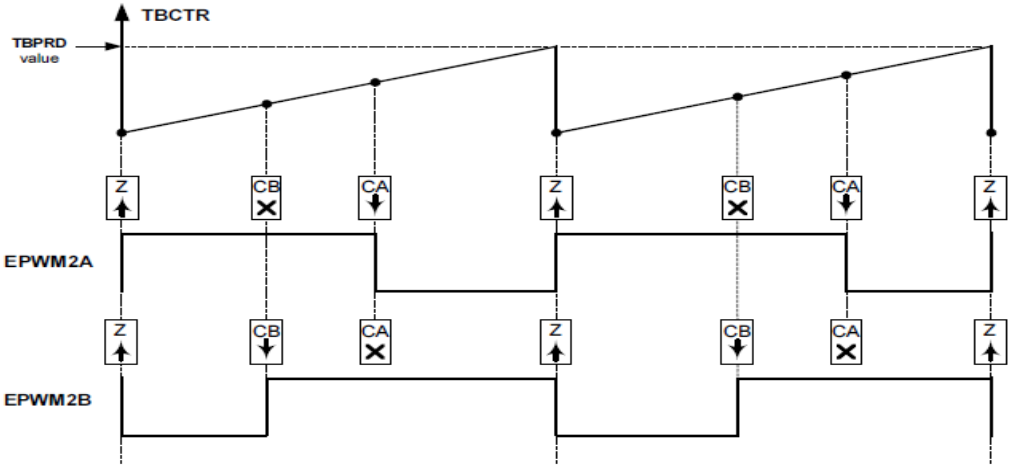
- Postavi PWM2A na događaj TBCTR=0 (Z)

- Obriši PWM2A na događaj TBCTR=CMPA
- Postavi PWM2B na događaj TBCTR=0 (Z)
- Postavi PWM2B na događaj TBCTR=CMPB [6]

Programski kod za ovu konfiguraciju prikazan je na slici 4.7, dok je stvaranje signala prikazano slikom 4.8.

```
// Postavljanje akcija
EPwm2Regs.AQCTLA.bit.ZRO = AQ_SET; // Postavi PWM2A na nulu
EPwm2Regs.AQCTLA.bit.CAU= AQ_CLEAR; // Obriši PWM2A na događaj
A, uzlazno brojanje
EPwm2Regs.AQCTLB.bit.ZRO = AQ_CLEAR; // Obriši PWM2B na nulu
EPwm2Regs.AQCTLB.bit.CBU = AQ_SET; // Postavi PWM2B na događaj
B,uzlazno brojanje
```

Slika 4.7 Programski kod akcijskog kvalifikacijskog modula za dva asimetrična PWM-a [6]



Slika 4.8 Princip stvaranja asimetričnih PWM signala [6]

Frekvencija PWM signala za uzlazni mod brojača računa se po sljedećim izrazima:

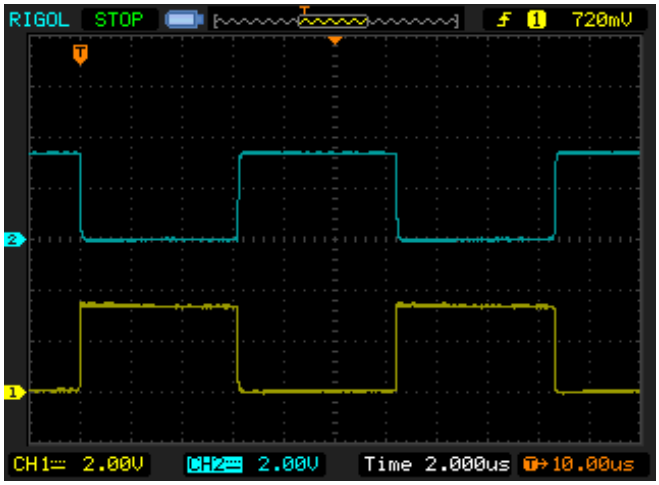
$$T_{PWM} = (TBPRD + 1) * T_{TBCLK} \quad (4\text{-}3)$$

$$F_{PWM} = 1/(T_{PWM}) \quad (4\text{-}4)$$

Gdje je TBPRD period postavljen početno u programu na vrijednost 1000, a period takta procesora T_{TBCLK} je 1/80 MHz. Na osnovu izraza 4-3 i 4-4 generirana PWM frekvencija (F_{PWM}) ima vrijednost 80 kHz. Valni oblici se promatraju preko osciloskopa koji se spaja na EPWM2A i EPWM2B pinove. Oblik PWM valnog oblika može se mijenjati mijenjanjem koda akcijskog kvalifikacijskog modula dok se frekvenciju mijenja promjenom vrijednosti perioda. Širina impulsa za svaki PWM kanal može se neovisno mijenjati iz prozora za promatranje varijabli mijenjanjem vrijednosti duty_cycleA i

duty_cycleB. Također se mora paziti da te vrijednosti budu manje od vrijednosti perioda. Taj dio programskog koda izgleda isto kao i kod simetrične PWM modulacije.

Mjerenje se radi na osciloskopu, a kako je postavljena vrijednost perioda 1000, frekvencija moduliranih valova je 80 kHz. Vrijednosti širine impulsa postavljena je na 500 te će modulirani signali biti pola perioda uključeni, što je prikazano na slici 4.9.

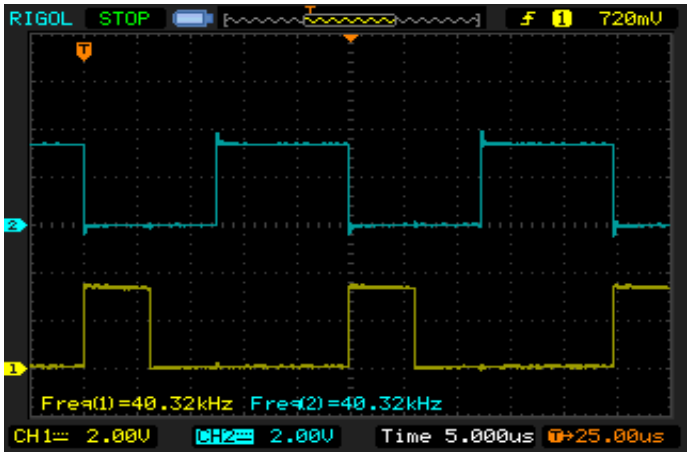


Slika 4.9 Asimetrični PWM signali frekvencije 80 kHz

Kao što je vidljivo na slici 4.9, brojač TBCTR za kanal EPWM2A, koji je prikazan žutom bojom, postavlja u nuli signal na određeni napon te broji do postavljenog perioda 500. U tome trenutku on briše taj signal te nastavlja brojiti do TBPRD perioda 1000. Tada resetira brojač na nulu i ponovo počinje isti postupak. Kanal EPWM2B koji je predstavljen plavom bojom na osciloskopu predstavlja komplementarni modulirani signal EPWM2A signalu, te je dobiven na sličan način, samo što sada brojač radi sve suprotno. U trenutku kad počinje brojati postavlja signal na nulu te broji uzlazno do postavljenog perioda u prozoru za namještanje varijabli. Kada je dosegao tu vrijednost od 500 postavlja signal i nastavlja uzlazno brojiti do zadanog TBPRD perioda 1000, te se tada resetira, postavlja na nulu i briše taj signal. Nakon toga započinje isti postupak ponovo.

U sljedećem mjerenju postavljena vrijednost TBPRD perioda je 2000 što će automatski rezultirati promjenom frekvencije oba signala na 40 kHz. U prozoru za gledanje varijabli promjenjene vrijednosti za upravljanje širinom impulsa iznose 500 za kanal EPWM2A te 1000 za kanal EPWM2B.

Iz slike 4.10. je vidljivo da su dobijena dva neovisno modulirana signala, gdje kanal EPWM2A (prikazan žutom bojom) vodi četvrtinu perioda, dok kanal EPWM2B (prikazan plavom bojom) vodi pola perioda te je komplementaran prvom signalu.



Slika 4.10 Asimetrični PWM signali neovisno modulirani

4.3. Asimetrični PWM sa mrtvim vremenom (DeadBand PWM)

4.3.1. Svrha programa

Ovaj program konfigurira EPWM2 vanjsku jedinicu da generira dva komplementarna (asimetrična) valna oblika, gdje će jedan od njih imati određeni iznos kašnjenja odnosno mrtvog vremena kod uzlaznog i silaznog brida. Postavka pinova ista je kao i kod simetričnog i asimetričnog PWM-a te je prikazana tablicom 4.3. [6]

Tablica 4.3 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

Nema spoja	Osciloskop

Nakon učitavanja programa u mikrokontroler, u prozor za promatranje varijabli dodaju se sljedeće varijable:

- RED_delay (Ova varijabla namiješta kašnjenje uzlaznog brida).
- FED_delay (Ova varijabla namiješta kašnjenje silaznog brida).

Početno su obje varijable namještene na 375 ns.

4.3.2. Opis rada programa

DeadBand podmodul podržava neovisno namještanje vrijednosti varijabli kašnjenja za uzlazni i silazni brid. Kontrola i iznos kašnjenja programira se korištenjem sljedećih registara gdje hexadecimalni broj predstavlja adresu registra:

- DBCTL (0x000F) Dead-Band kontrolni registar
- DBRED (0x0010) Dead-Band brojački registar zadužen za kašnjenje uzlaznog brida
- DBFED (0x0011) Dead-Band brojački registar zadužen za kašnjenje silaznog brida

Navedeni registri su deset bitni, a vrijednosti DBRED i DBFED registra predstavljaju broj vremenskih perioda takta procesora za koje je uzlazni ili silazni brid zakašnjen. Izrazi za izračun kašnjenja uzlaznog RED (Rising Edge Delay) i silaznog FED (Falling Edge Delay) brida su sljedeći:

$$FED = FED_delay * T_{TBCLK} \quad (4\text{-}5)$$

$$RED = RED_delay * T_{TBCLK} \quad (4\text{-}6)$$

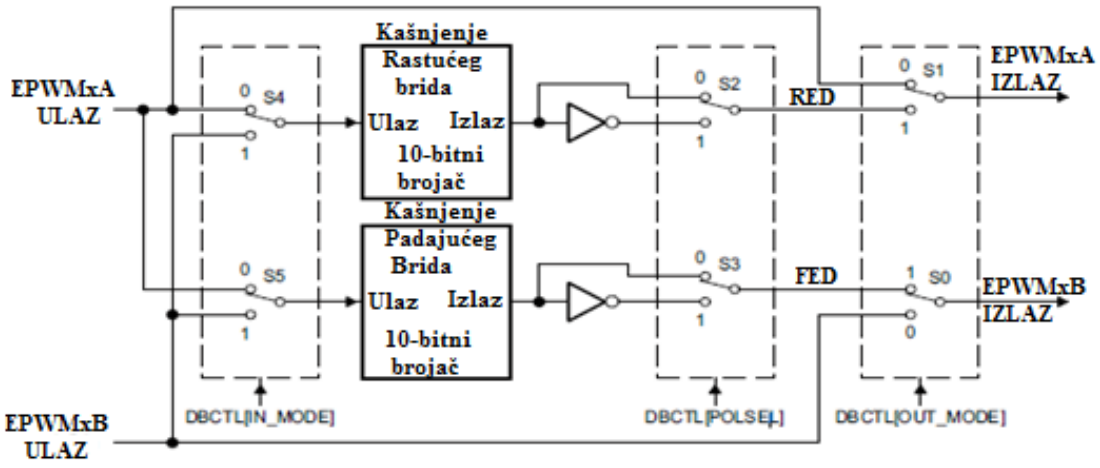
U programu su RED_delay i FED_delay postavljeni na 30 dok je T_{TBCLK} period jednog takta procesora i iznosi 1/80 MHz. Stoga će kašnjenja uzlaznog i silaznog brida u ovom primjeru biti 0.375μs. Dead-band podmodul ima dvije grupe nezavisnih opcija namještanja kako je prikazano na slici 4.11, a to su:

1. Namještanje odabirom ulaza gdje su ulazni signali u dead-band podmodul EPWMxA i EPWMxB, izlazni signali iz akcijskog kvalifikacijskog modula. Na slici 4.11. prikazani su EPWMxA ulazi i EPWMxB ulazi. Korištenjem DBCTL (ulaznog moda) kontrolnih bitova, izvor signala za svako kašnjenje bilo uzlazno ili silazno namješta se na sljedeći način:

- EPWMxA je izvor za uzlazno i silazno kašnjenje te se on koristi u ovom programu. To je ujedno i početno postavljen mod.
- EPWMxA je izvor za kašnjenje padajućeg brida, dok je EPWMxB izvor za kašnjenje rastućeg brida.
- EPWMxA je izvor za kašnjenje rastućeg brida, dok je EPWMxB izvor za kašnjenje padajućeg brida.
- EPWMxB je izvor za kašnjenje rastućeg i padajućeg brida.

2. Namještanje odabirom izlaza, gdje je izlazni mod konfiguriran DBCTL (izlazni_mod) kontrolnim bitovima. Ti bitovi određuju da li je kašnjenje ulaznog brida, silaznog brida, ni silaznog ni uzlaznog ili oba brida pridruženo ulaznim signalima.

Također postoji i kontrola polariteta DBCTL (POLSEL) koja omogućuje hoće li kašnjenje signala padajućeg ili rastućeg brida ili oba brida invertirati, prije nego što se pošalje na izlaz Dead-Band podmodula. [6]



Slika 4.11 Konfiguracijske opcije mrtvo-zonskog podmodula [6]

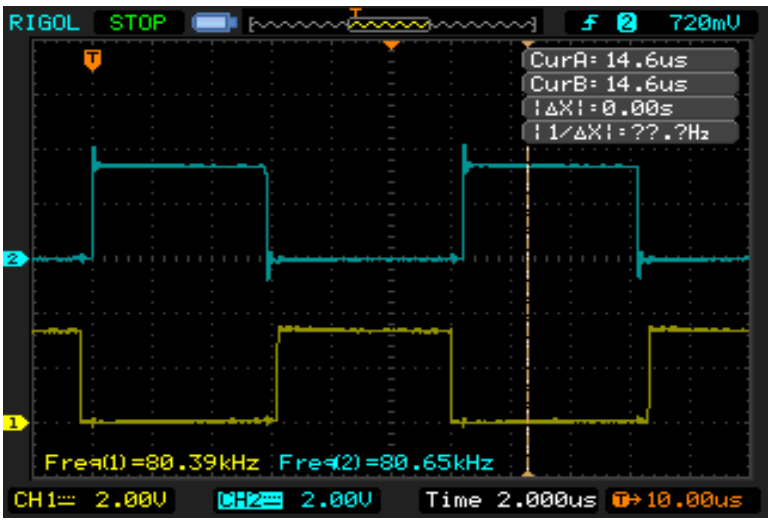
U ovom primjeru promatraju se PWM signali spajanjem EPWM2A i EPWM2B pinova preko osciloskopa, gdje će EPWM2A biti predstavljen žutom bojom te će imati kašnjenja uzlaznog i silaznog brida od 0.375 μs. EPWM2B kanal biti će predstavljen plavom bojom i on će predstavljati komplementarni (asimetrični) PWM valni oblik kanalu EPWM2A te on neće imati kašnjenja uzlaznog i silaznog brida. Kašnjenja uzlaznog i silaznog brida za EPWM2A kanala moći će se neovisno namještati iz prozora za gledanje varijabli (watch window) koji je prikazan na slici 4.12.

Local (1) | Watch (1)

Name	Value	Address	Type	Format
(x)= RED_delay	30	0x00008C08@Data	unsigned int	Natural
(x)= FED_delay	30	0x00008C09@Data	unsigned int	Natural
<new>				

Slika 4.12 Prikaz varijabli za mijenjanje rastućih i padajućih bridova PWM-a

Kako je TBPRD period postavljen na 1000, frekvencija PWM signala iznositi će 80 kHz kao što je prikazano na slici 4.13.



Slika 4.13 Prikaz reguliranja mrtve zone kod PWM signala prikazanog žutom bojom

Također je vidljivo da kašnjenje uzlaznog i silaznog brida iznosi oko jednu petinu vremenske baze (jedne kockice) na osciloskopu, koja je 2 µs. Kad to podijelimo dolazimo taman otprilike do kašnjenja uzlaznog i silaznog brida od 0.375 µs.

4.4. Okidni PWM

4.4.1. Svrha programa

Ovaj program konfigurira PWM1 i koristi GPIO-18 pin konfiguriran kao GPIO izlaz sa stanjima 0 ili 1 kojega dovodimo na ulaz PWM Trip okidne zone (TZ1-GPIO12 pin). Korisnik može konfigurirati TZ1 okidni signal kao jedan pokušaj OSHT (One-Shot Trip) ili ciklus po ciklus CBC (Cycle-By-Cycle) okidanje. Nakon što se trip (okidni) događaj dogodi, jedan PWM signal otići će u nulu, dok će drugi poprimiti stalni iznos od 3.3 V. [6]

Tablica 4.4 Postavka sklopovlja mikrokontrolera za efekt trip zone

1 ADC-A6 COMP3(+VE)	2 **ADC-A2** **COMP1 (+VE)**	3 **ADC-A0**	4 **3V3**
5 ADC-A4 COMP2 (+VE)	6 **ADC-B1**	7 **EPWM-4B** **GPIO-07**	8 **TZ1** **GPIO-12**
9 SCLA GPIO-33	10 **ADC-B6** **COMP3(-VE)**	11 **EPWM-4A** **GPIO-06**	12 **ADC-A1**
13 SDAA GPIO-32	14 **ADC-B0**	15 **EPWM-3B** **GPIO-05**	16 **5V0** **(DisabledbyDefault)**
17 EPWM-1A GPIO-00	18 **ADC-B4** **COMP2 (-VE)**	19 **EPWM-3A** **GPIO-04**	20 **SPISOMIA** **GPIO-17**
21 EPWM-1B GPIO-01	22 **ADC-A5**	23 **EPWM-2B** **GPIO-03**	24 **SPISIMOA** **GPIO-16**
25 SPISTEA GPIO-19	26 **ADC-B2** **COMP1 (-VE)**	27 **EPWM-2A** **GPIO-02**	28 **GND**
29 SPICLKA GPIO-18	30 **GPIO-34** **(LED)**	31 **PWM1A-DAC** **(Filtered)**	32 **GND**

Osciloskop	**Spojna žica**	**Nema konekcije**

Nakon učitanja programa u mikrokontroler, u prozor za promatranje i promjenu varijabli postavljaju se sljedeće varijable:

- one_shot. Ova varijabla konfigurira TZ (Trip Zone) akciju ili kao jedan pokušaj ili ciklus po ciklus. Ako je one_shot=1, tada je TZ1 konfiguriran kao jedan pokušaj trip okidni događaj. Ako je one_shot=0, tada je TZ1 konfiguriran kao ciklus po ciklus trip okidni događaj.
- GpioDataRegs.GPATOGGLE.bit.GPIO18. Ovaj registar namješta vrijednost izlaza GPIO18 pina, postavljanjem u 0 ili 1.

4.4.2. Opis rada programa

Trip-zonski signal koristi se da bi zaustavio sklopovlje, a može se koristiti i za stalno generirane PWM signale. Trip-zonski signali na pinovima TZ1 do TZ3 su aktivni u niskom stanju odnosno nuli. Kada jedan od tih pinova ode u nulu, to govori da se trip događaj dogodio. Svaki EPWM modul može se individualno konfigurirati da koristi ili ignorira svaki pin trip zone. Odabir trip-zonskih signala koje će koristiti odabrani EPWM modul određeno je TZSEL (Trip-Zone Select Register) registrom. U ovisnosti o trip

događaju moguće su sljedeće EPWM izlazne akcije koje su definirane u TZCTL (Trip-Zone Control Register) registru:

- Visoka impendancija
- Dovedi u visoko stanje
- Dovedi u nisko stanje
- Ne mijenjaj ništa

Svaki TZn ulazni pin može biti pojedinačno konfiguriran da radi ciklus po ciklus ili kao jedan pokušaj trip okidanja, za određeni EPWM modul. DCAEVT1 i DCBEVT1 događaji mogu biti konfigurirani da budu direktno jedan pokušaj trip okidnog događaja EPWM modulu. Isto tako DCBEVT1 i DCBEVT2 događaji mogu biti konfigurirani da budu ciklus po ciklus okidni trip događaj EPWM modulu. Ova konfiguracija određena je sa TZSEL[DCAEVT1/2], TZSEL[DCBEVT1/2], TZSEL[CBCn] i TZSEL[OSHTn], što su registri digitalne usporedbe trip zone gdje n predstavlja broj TZ signala odnosno njegovog pina. U nastavku će biti opisani trip događaji:

- Ciklus po ciklus trip:

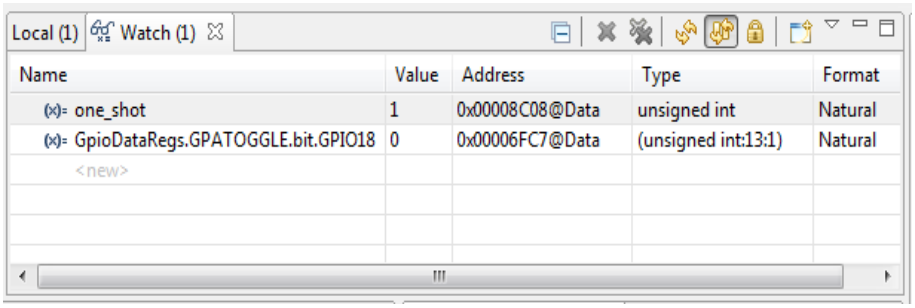
Kada se dogodi ciklus po ciklus trip događaj, akcije definirane u TZCTL registru automatski se prenose na izlaze EPWM modula. Ako je postavljena zastavica u TZFLG (Trip-Zone Flag Register) registru, generiran je prekid EPWMx_TZINT koji mora biti i omogućen u TZEINT (Trip-Zone Enable Interrupt Register) registru. Ako smo omogućili taj prekid te ako su DCAEVT2 i DCBEVT2 odabrani kao trip izvori u TZSEL registru, nije potrebno posebno aktivirati DCAEVT2 i DCBEVT2 u TZEINT registru jer DC događaj okida prekid kroz ciklus po ciklus mehanizam. Specificirana stanja na ulazima su automatski obrisana kada EPWM vremensko-bazni TBCTR dođe na nulu. Stoga u ciklus po ciklus modu, trip događaj se briše ili resetira svaki PWM ciklus. TZFLG zastavica će ostati dok se ručno ne obriše u TZCLR registru. Ako je ciklus po ciklus trip događaj još uvijek prisutan kada je zastavica obrisana, opet će biti ponovo postavljen.

- OSHT (One Shot Trip)

Kada se dogodi jedan okidni trip događaj, akcija specificirana u TZCTL registru automatski se prenosi na izlaze EPWM modula. Dodatno, jedan pokušaj (one shot) zastavica je postavljena u (TZFLG[OST]) flag registru te je EPWMx_TZINT prekid generiran, i isto mora biti omogućen u TZEINT registru. Jedan pokušaj stanje mora biti izbrisano ručno brisanjem TZCLR[OST] bita. Ako je jedan pokušaj prekid omogućen sa

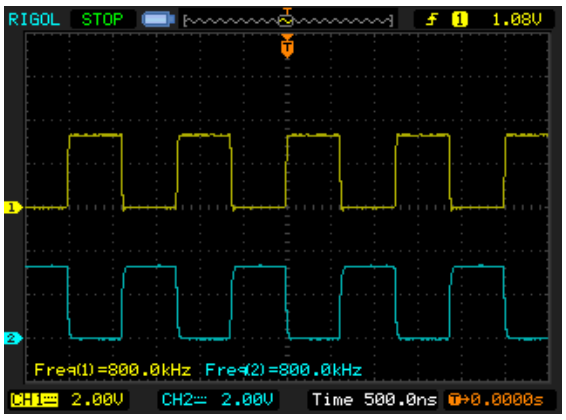
TZEINT registrom te su DCAEVT1 i DCBEVT1 odabrani kao trip izvori sa TZSEL registrom, nije potrebno omogućiti DCAEVT1 I DCBEVT1 prekide u TZEINT registru, jer DC događaj okida prekide preko OSHT mehanizma. [1]

U ovom programu povezuje se TZ1(GPIO12) pin sa GPIO18 pinom spojnom žicom, te se nakon toga radi pokretanje programa. Poslje toga mijenja se stanje TZ1 pina u 0 da se postigne efekt trip zone. Za tu namjenu koristiti će se vrijednost regista GPIO18 pina. Mijenjanje stanja radi se iz prozora za promatranje varijabli slika 4.14.



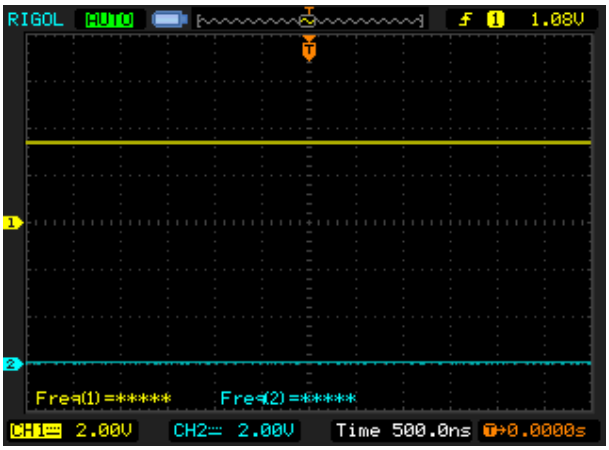
Slika 4.14 Prikaz varijabli za namiještanje trip zone

Prilikom pokretanja programa generirat će se dva asimetrična PWM signala na EPWM-1A i EPWM-1B pinu, što je prikazano slikom 4.15. Vidljivo je da u tom trenutku ne postoje nikakve promjene moduliranih signala, zbog toga što je vrijednost TZ1 signala u jedinici.



Slika 4.15 Dva asimetrična PWM signala bez efekta trip zone

Kako je stanje TZ1 određeno sa GPIO18, u prozoru za promjenu i promatranje varijabli korisnik mijenja njegovo stanje iz jedinice u nulu. Tim postupkom se dogodio okidni trip događaj, jer će TZ1 signal otići u nulu. Sve to utjecalo je na promjenu moduliranih signala, ali na onaj način kako je to definirano u TZCTL registru. Iz slike 4.16 je vidljivo da će signal EPWM-1A otići u visoko stanje na 3.3V, dok će EPWM-1B otići u nisko stanje odnosno nulu.



Slika 4.16 Efekt trip zone

4.5. Filtrirani visokorezolucijski PWM

4.5.1. Svrha programa

Ovaj program pokazuje kako proizvesti različite vrijednosti napona koristeći jednostavnu DAC (Digital-To-Analog Converter) digitalno analognu pretvorbu oblikovanu filtriranjem izlaza na rastućem bridu iz visoko rezolucijskog PWM-a. [6]

Tablica 4.5 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 HRPWM1A-DAC (Filtered)	32 GND

Spoj sa multimetrom	Nema spoja

Nakon učitanja programa u pločicu u prozor za promatranje i promjenu varijabli dodaju se sljedeće varijable:

- duty_cycle (decimalni format). Ova varijabla namiješta širinu impulsa PWM-a, da bi se mogao mijenjati DC napon na izlazu iz DAC.

4.5.2. Opis rada programa

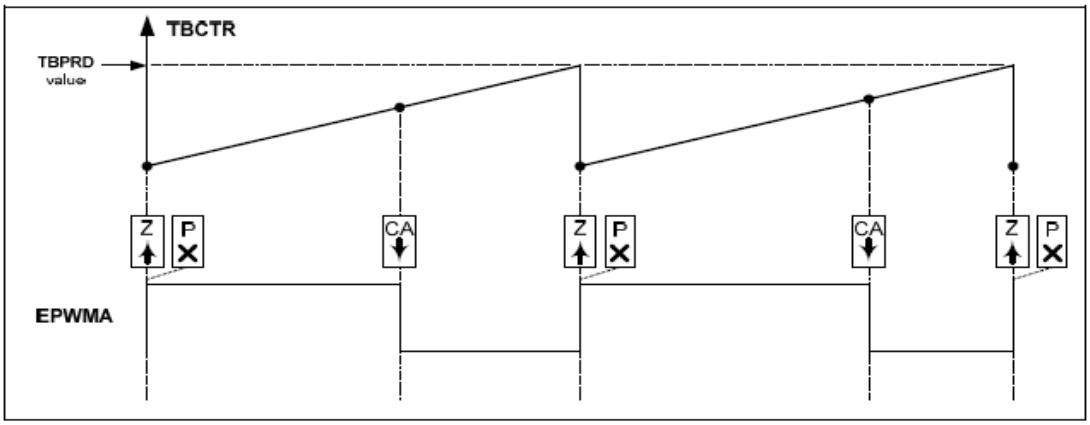
U ovome programu prvi korak je konfiguriranje PWM vanjske jedinice koja radi u uzlaznom brojačkom modu. U uzlaznom brojačkom modu PWM brojač TBCTR, koji je direktno povezan sa periodom takta procesora, broji od nule do postavljene vrijednosti perioda TBPRD. Kad god brojač dostigne vrijednos perioda resetira se na nulu i počinje brojati iznova. Oblik generiranog PWM signala određen je načinom na koje su postavljene akcije akcijskog kvalifikacijskog modula na određene događaje. Za ovaj projekt sljedeće akcije-događaji su korištene:

- Postavi PWM na događaj TBCTR=0(Z)
- Obriši PWM na događaj TBCTR=CMPA [6]

Programski kod za ovu konfiguraciju prikazan je na slici 4.17, a stvaranje signala slikom 4.18.

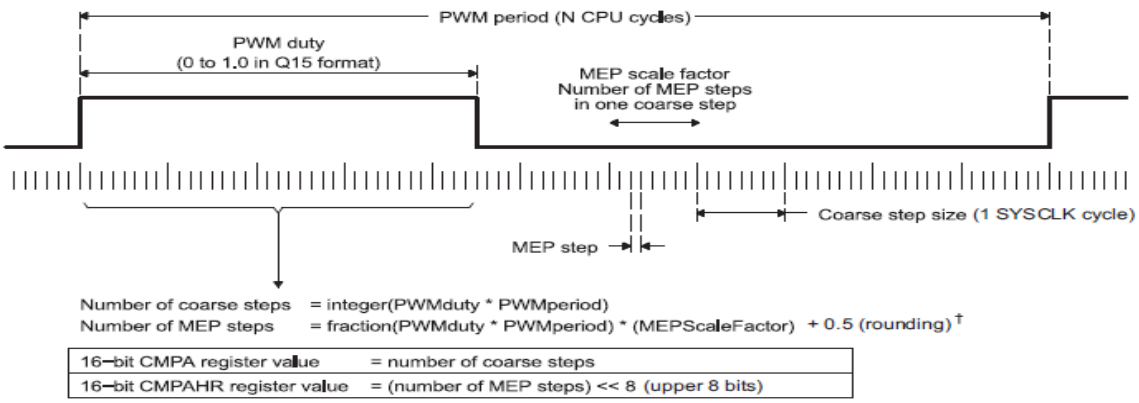
```
// postavljene akcije
EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET; // Postavi PWM1A na nulti
događaj
EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR; // Obriši PWM1A na uzlazno
brojački CompareA događaj
```

Slika 4.17 Programski kod akcijskog kvalifikacijskog modula



Slika 4.18 Prikaz stvaranje PWM signala u uzlaznom brojačkom modu [6]

Korištenjem HRPWM modula povećava se vremenska rezolucija konvencionalnog PWM-a. HRPWM obično se koristi kad PWM rezolucija padne ispod 9-10 bitova. Premda je svaka aplikacija drugačija, obične PWM operacije ispod 250 kHz vjerovatno neće zahtijevati HRPWM. HRPWM je baziran na mikro rubnom pozicioniranju odnosno MEP (Micro Edge Positioner) tehnologiji. Preciznost vremenskog koraka reda je 150 ps. Slika 4.19 prikazuje odnos između vremenskog perioda sistemskog sata (coarse system clock) i rubne pozicije u pogledu MEP koraka, koji su kontrolirani 8 bitnim područjem s usporedbenim visokorezolucijskim A registrom CMPAHR (Compare A Extension Register). [1]



Slika 4.19 Operacijska logika koju koristi MEP tehnologija [1]

HRPWM je konfiguriran na način da MEP logika koristi CMPAHR registar kako bi kontrolirala poziciju rastućeg brida moduliranog signala. Ovo pruža korisniku finu kontrolu rubnog položaja. Kod za ovu konfiguraciju prikazan je na slici 4.20.

```
// Postavi visokorezolucijske registre
EALLOW; // Otključaj kontrolne registre za unos
EPwm1Regs.HRCNFG.all = 0x0; // Inicijaliziraj HRPWM controlne
registre
EPwm1Regs.HRCNFG.bit.EDGMODE = HR_REP; // Omogući
visokorezolucijsku kontrolu na rastućem bridu
EDIS; //Zaključaj kontrolne registre
```

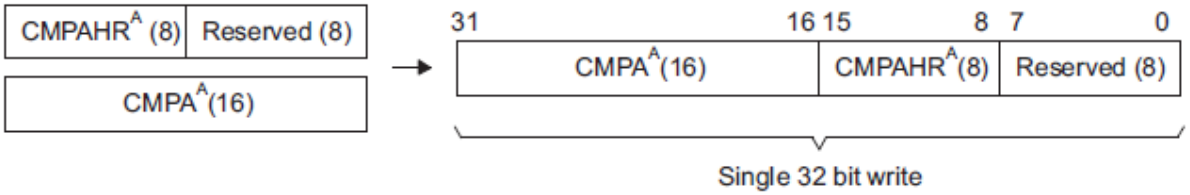
Slika 4.20 Programski kod za omogućavanje visokorezolucijskih registara i kontrolu rastućeg brida [6]

Korisnička kontrola HRPWM omogućena je upotrebom HRPWM proširenog registra. Taj registar namijenjen je da omogući korisniku pisanje 32-bitne riječi za postavljanje CMPA i CMPAHR registara. Upotreba ovog registra omogućava veliku kontrolu rubnog

područja kroz cijeli period vala. Osam nižih bitova u registru su rezervirani, stoga je vrijednost pomaknuta u lijevo za 8 bitova što je prikazano na slici 4.22, te programskim kodom na slici 4.21.

```
EPwm1Regs.CMPAM.all = duty_cycle<<8; //Ažuriraj usporedbeni A
registar koristeći uniju CMPA i CMPAHR registara.
//Ova vrijednost promijenit će nivo PWM1A_DAC izlaza.
//Vrijednost ove varijable mora biti između 0 i 1E00h
(period*65536)>>8
// Osam nižih bitova se ne računa
```

Slika 4.21 Programski kod za ažuriranje usporedbenog A registra [6]



Slika 4.22 Brojački usporedbeni visokorezolucijski A registar [6]

Nakon učitanja programa u mikrokontroler, u prozoru za namještanje varijabli dodaje se varijabla duty_cycle. Kako se tom varijablom mijenja širina impulsa istovremeno se mijenja i izlazni DC napon. Početno je u programu duty_cycle namješten na pola perioda što odgovara decimalnom broju 5120. Vrijednost napona izmjerena multimetrom za pola perioda uključenosti filtriranog vala iznositi će 1.65V. Mijenjajući vrijednost perioda vala od 0 do punog perioda 10240 moći će se vrlo fino namještati vrijednost izlaznog DC napona.

4.6. Uspoređivanje analognih vrijednosti komparatorom

4.6.1. Svrha programa

Ovaj program konfigurira EPWM1A (analogni modulirani signal), komparator i unutarnji DAC (analogni napon). Korisnik može uspoređivati EPWM1A analognu vrijednost napona ili nekakav vanjski analogni napon sa unutarnjim DAC analognim naponom. U ovisnosti o stanju izlaza komparatora, LED dioda L2 na mikrokontroleru će zasvijetliti ili ne. [6]

Tablica 4.6 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

	Spojna žica	**Nema spoja**

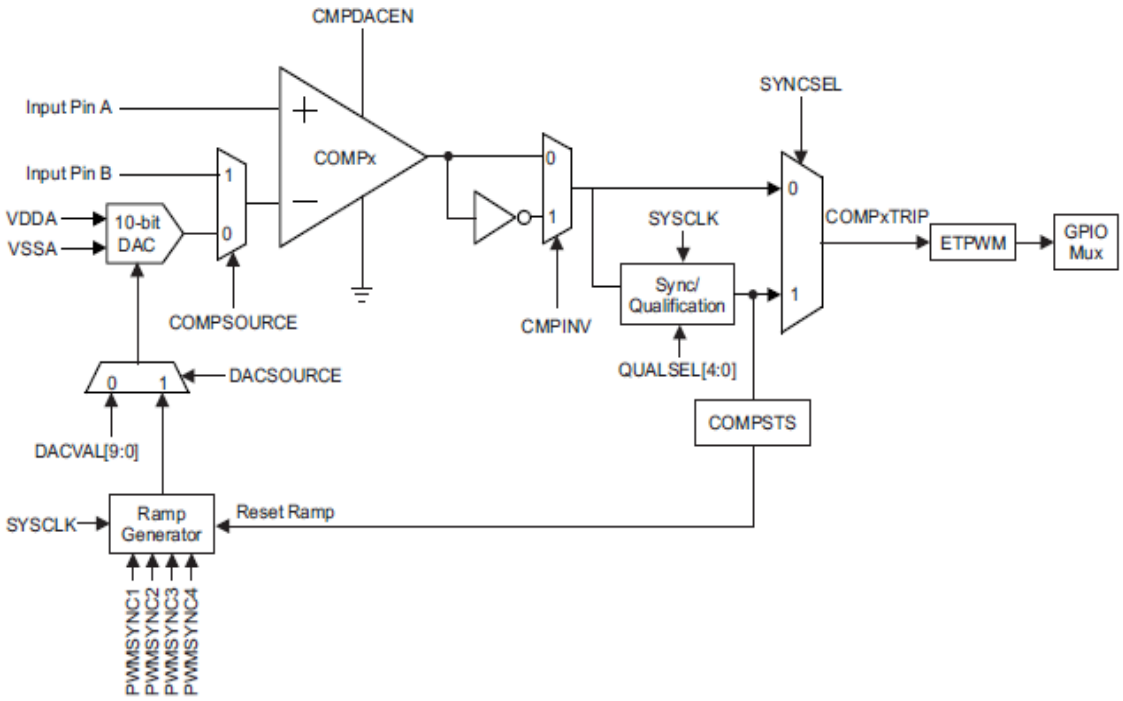
Što se tiče softverske postavke nakon učitanja programa u pločicu, u prozor za promatranje varijabli dodaju se sljedeće varijable:

- dac_value. Ova varijabla namiješta unutarnji analogni napon. Format pisanja broja je decimalni te se piše u rasponu od 0 do 1024 što odgovara rasponu napona od 0 do 3.3V.
- duty_cycle_A. Ova varijabla postavlja širinu impulsa PWM1A-DAC te se zbog toga na njegovom izlazu mijenja vrijednost DC (istosmjernog) napona. Vrijednost širine impulsa mora biti manja od vrijednosti perioda.

4.6.2. Opis rada programa

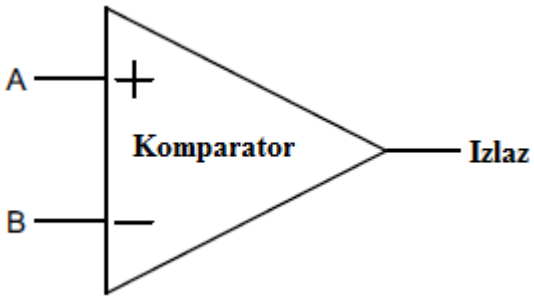
Ovim programom unutarnja analogna vrijednost napona je uspoređena s vanjskom vrijednošću DC napona koji se dovodi preko vanjskog izvora ili je proizveden preko PWM vanjske jedinice. Uređaj koji radi usporedbu između ta dva analogna napona zove se komparator te je pikazan na slici 4.23. Njegov modul sastoji se analognih ulaza, komparatora, izlaza te unutarnjeg DAC napona. On može izvršiti usporedbu između dva vanjska analogna napona ili jednog vanjskog analognog napona s unutarnjim DAC

naponom. Izlaz iz komparatora može biti nesinhroniziran ili sinhroniziran s periodom rada takta procesora te se dalje šalje na trip mehanizam PWM vanjskih jedinica, kao i na GPIO multipleksere.



Slika 4.23 Komparatorski blok dijagram [1]

Komparator u blok shemi je analogni modul te zbog toga njegov izlaz nije sinhroniziran sa frekvencijom rada takta procesora. Sam komparator prikazan je na slici 4.24, dok je njegova funkcija prikazana tablicom 4.7.



Slika 4.24 Komparator [1]

Tablica 4.7 Tablica istine komparatora

Naponi	Izlaz
Napon A> Napon B	1
Napon B>Napon A	0

Svaki blok komparatora sadrži 10-bitni DAC napon koji može biti korišten kao ulaz u invertirajući B ulaz komparatora te se koristi umjesto vanjskog analognog napona u ovom primjeru. U programu postavljena vrijednost DAC napona je manja od napona PWM1A-DAC. Ta DAC vrijednost postavljena je na 310 što je gledajući po formuli (310/1024)x3.3V približno jednako 1V. PWM1A-DAC raditi će u uzlaznom brojačkom modu te će vrijednost njegovog TBPRD perioda biti 40, što je prisjetimo li se uzlaznog načina rada PWM-a i njegov ukupni period. Njegova akcija-događaj postavit će se na period od 20 što će značiti da će signal biti 50% vremena uključen, odnosno isključen te će od maksimalnog napona koji je 3.3V, davati pola što je 1.65V. Napon DAC zbog toga će biti manji od PWM1A-DAC napona te će komparator biti u stanju 1, odnosno led lampica koja predstavlja pin GPIO34 neće svjetliti. Ova dva napona namještaju se iz prozora za promatranje varijabli, što je omogućeno programskim kodom na slici 4.25.

```
for(;;) // Beskonačna petlja
{
Comp1Regs.DACVAL.bit.DACVAL =dac_value; //Promijeni(unutarnji)DAC
napon
EPwm1Regs.CMPA.half.CMPA = duty_cycle_A; //Promijeni DC naponski
nivo na PWM1A-DAC izlazu
```

Slika 4.25 Programski kod za mijenjanje unutarnjeg napona te napona na EPWM1A izlazu [6]

U daljnjem razmatranju postoje dvije mogućnosti. Ili će se smanjivati PWM1A napon mijenjanjem širine impulsa ili povećavati DAC napon iz prozora za gledanje i namještanje varijabli. Onoga trenutka kad DAC napon bude veći od PWM1A napona, komparator će otići u stanje 0. Tada će se aktivirati DCEVT2 događaj koji radi ciklus po ciklus i on je trip izvor (odnosno isključuje) PWM1A-DAC napon. U istome trenutku kad se to dogodi zasvjetliti će i LED L2 dioda crvenom bojom.

4.7. Okidni PWM generiran uporedbom analognih signala

4.7.1. Svrha programa

Svrha ovog projekta je postavljanje PWM izlaznog signala u visoko ili nisko stanje temeljeno na usporedbi vanjskog analognog napona i unutrašnjeg DAC analognog napona. [6]

Tablica 4.8 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

Osciloskop	Vanjsko DC napajanje<(3.3V)	Nema spoja

Nakon što je program učitan u pločicu, sljedeće varijable se postavljaju u prozor za promatranje i mijenjanje varijabli:

- dac_value. Ova varijabla namiješta DAC napon od 0 do 1024 u decimalnom formatu. To je jednako rasponu napona od 0 do 3.3V.
- trip. Ova varijablu se koristi da konfigurira trip okidni događaj. Kad je konfigurirano ciklus po ciklus trip okidanje onda je 0, a ako je konfigurirana kao jedan pokušaj trip okidanja onda je 1.

4.7.2. Opis rada programa

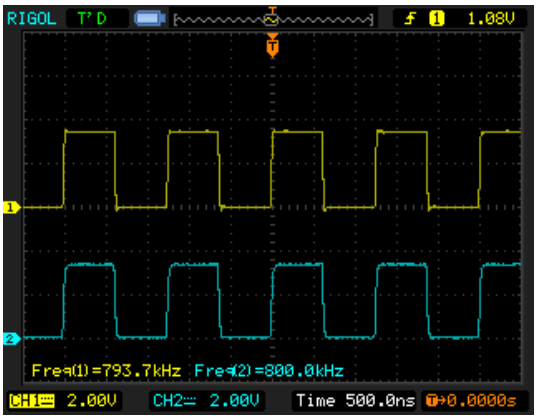
Ovaj program konfigurira EPWM1, komparator1 i unutrašnji DAC napon. Za razliku od programa Okidni PWM gdje su se koristili TZ pinovi kao trip zonski signali, ovdje će izlaz iz komparatora biti trip zonski signal. Komparator će raditi usporedbu vanjskog analognog napona i unutarnjeg DAC napona. Početno će se u programu namjestiti da je unutarnji DAC napon manji od vanjskog analognog napona te će EPWM1 normalno generirati signale. Nakon toga povećava se DAC napon da bude veći od

vanjskog analognog napona. Druga mogućnost je da se smanji vanjski napon na vrijednost manju od DAC vrijednosti napona. U trenutku kad DAC napon postane veći od vanjskog analognog napona komparator će otići u nisko stanje odnosno 0. To će aktivirati DCAEVT1, DCBEVT1, DCAEVT2 i DCBEVT2 događaje koji će zaustaviti PWM generaciju signala. EPWM1 će reagirati na DCAEVT1 i DCBEVT1 kao jedan pokušaj trip događaj, dok će na događaje DCAEVT2 i DCBEVT2 reagirati kao ciklus po ciklus trip događaj. DCAEVT1 i DCAEVT2 dovest će izlaz EPWM1A u visoko stanje, dok će događaji DCBEVT1 i DCBEVT2 dovesti izlaz EPWM1B u nisko stanje. Kada se u prozoru uspoređuje unutarnji napon za namiještanje i vanjski napon, može se promijeniti iznos unutarnjeg napona, na vrijednost veću od vanjskog napona, pa će komparator aktivirati trip okidne događaje. Mogućnost promjene varijabli prikazana je programskim kodom na slici 4.26.

```
for(;;) // Beskonačna petlja
{
Comp1Regs.DACVAL.bit.DACVAL =dac_value; //
Promijeni(unutarnju) DAC vrijednost
}
```

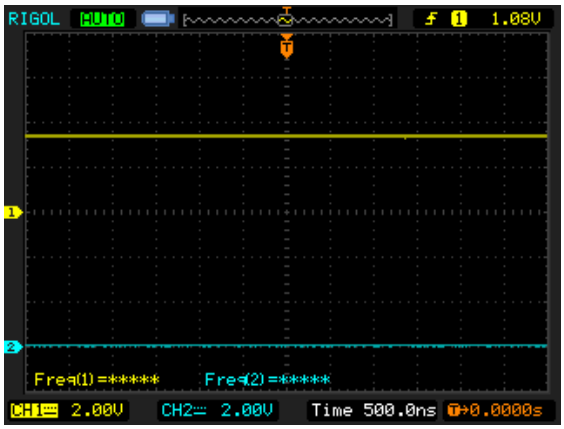
Slika 4.26 Programski kod za mijenjanje unutarnje DAC vrijednosti napona [6]

Nakon učitavanja programa u mikrokontroler, doveden je vanjski napon od 2V na pin ADC-A2. Kako se vrijednost unutarnjeg napona namiješta preko dac_value, on će početno biti namješten na vrijednost od 512, što je ekvivalentno naponu od 1.65V. S obzirom da je vanjski napon veći od unutarnjeg dva simetrična PWM signala će se normalno generirati, kako je prikazano slikom 4.27.



Slika 4.27 Prikaz generiranja dva simetrična PWM signala

Nakon što se povećana unutarnji napona na vrijednost veću od vanjskog napona komparator će aktivirati trip događaje. To će rezultirati da EPWM1A signal ode u visoko stanje, odnosno na napon od 3.3V, dok će EPWM1B signal otići u nulu.



Slika 4.28 Efekt trip zone temeljen na usporedbi dva analogna napona

4.8. Neprekidna ADC pretvorba

4.8.1. Svrha programa

Ovaj program demonstrira neprekidnu analognu digitalnu pretvorbu 11 ADC kanala koji su mogući na TMS320F28069 mikrokontroleru. Način spajanja analognih kanala prikazan je sljedećom tablicom pinova. [6]

Tablica 4.9 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

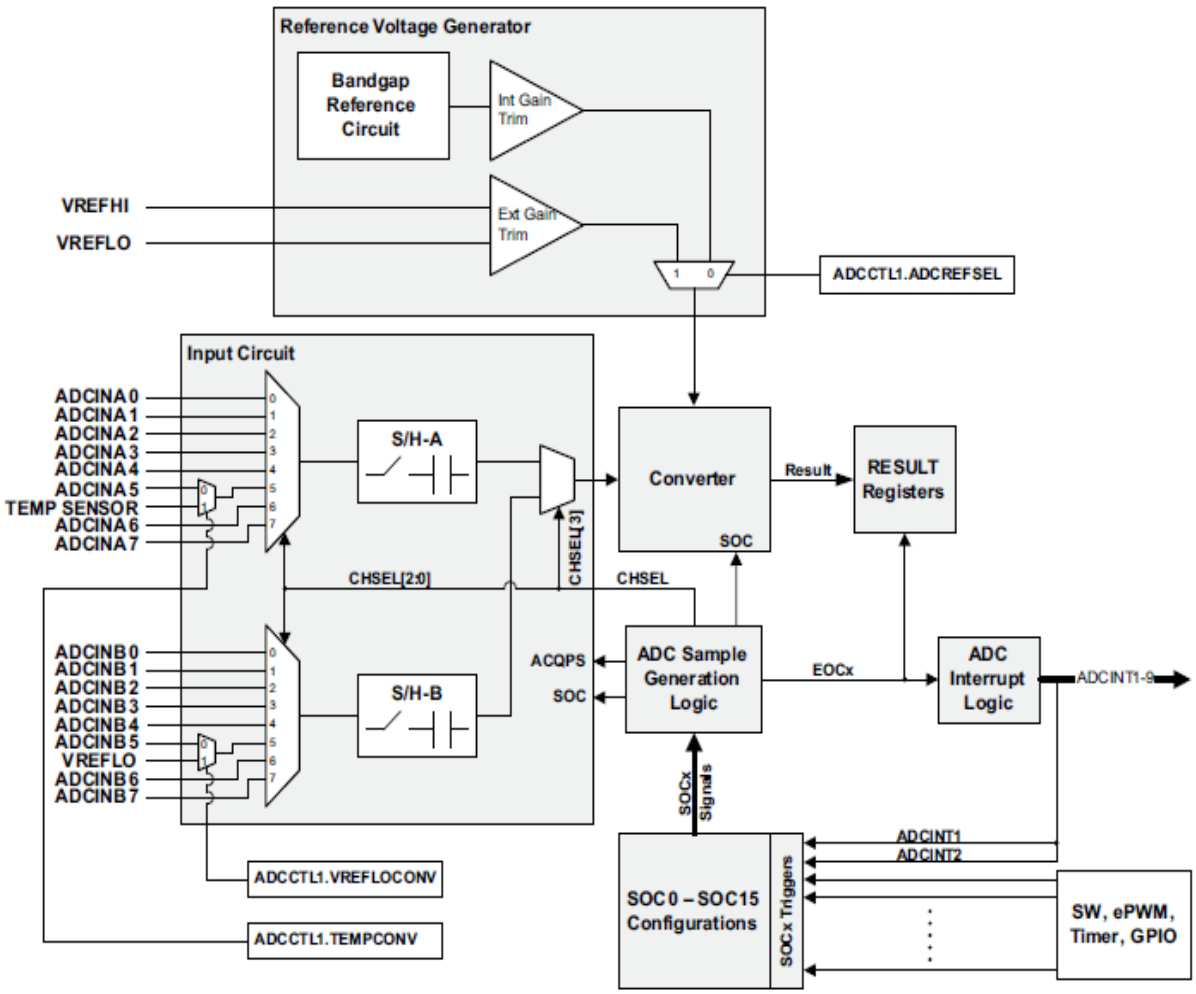
Nema spoja	Vanjski analogni napon(<3.3V)

Nakon što je program učitan u pločicu sljedeće varijable se dodaju u prozor za promatranje varijabli:

- AdcResults (format= hexadecimalni). Ovaj red pohranjuje pretvorene vrijednosti 11 ADC kanala na mikrokontroleru. Vrijednosti su u rasponu od 0x0000 do 0x0FFF, što je ujedno i ekvivalentno rasponu od 0 do 3.3V.

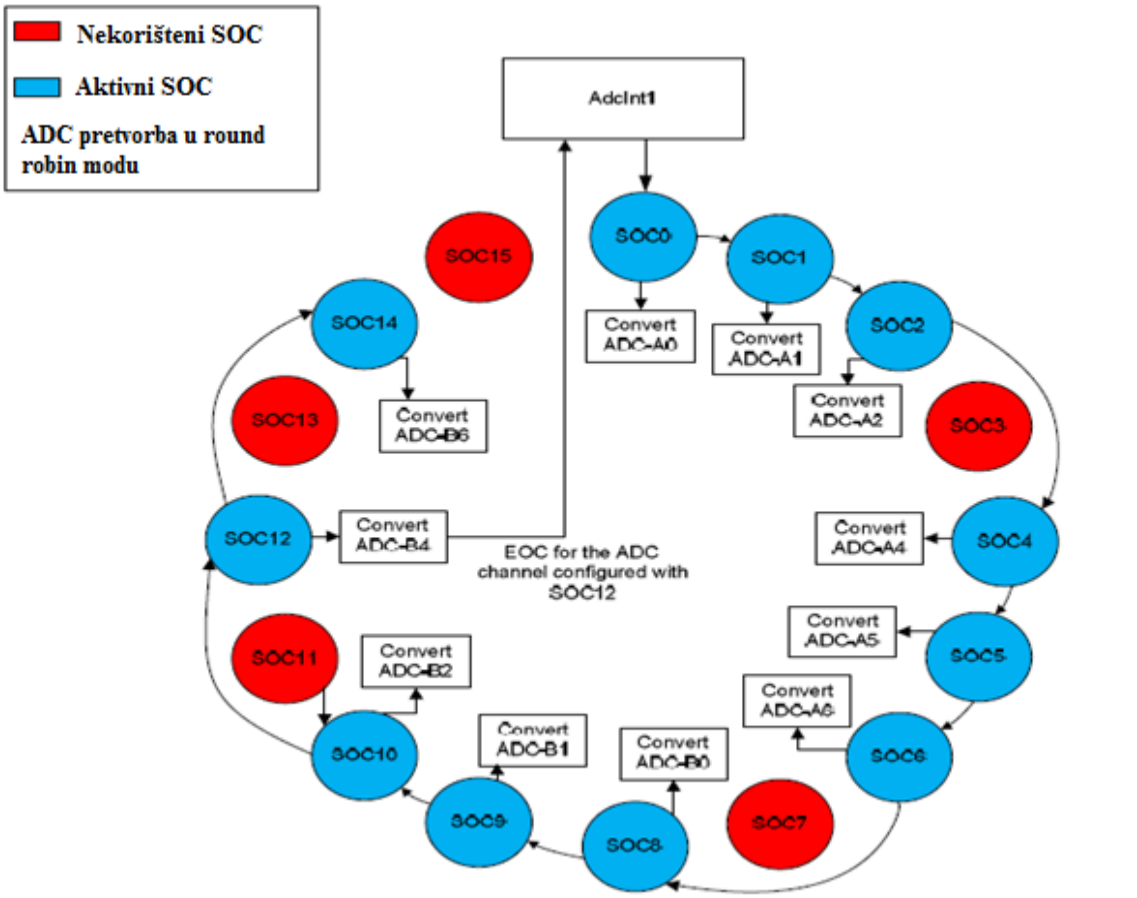
4.8.2. Opis rada programa

Jezgra ADC analogno digitalnog pretvornika sadrži jedan 12-bitni pretvornik koji je snabdijevan sa dva izvora uzrokovanih signala. ADC pretvarač TMS320F28069 mikrokontrolera nije baziran sekvencijalno. Princip pretvaranja bazira se na serijama pretvorbi određenih nekakvim događajem. Za početak svake pojedinačne pretvorbe zadužen je SOC (Start Of Conversion). Svaki SOC se može nezavisno konfigurirati i može imati bilo koju kombinaciju okidača za start, kanala za pretvorbu i veličinu prozora za sampliranje. ADC vanjska jedinica sadrži još regulatore napona, registre rezultata, prekidnu logiku te logiku za povezivanje sa drugim modulima koji određuju početak komunikacije mikrokontrolera. [1]



Slika 4.29 Blok dijagram ADC pretvarača [1]

Kako je ADC pretvarač konfigiruriran na principu SOC postoje različiti okidački mehanizmi za početak starta pretvorbe, a to su: EPWM, S/W (Software Imediate Start) odnosno trenutačni start programski aktiviran, vanjski signali, prekidi, itd. Ovim programom ADC prekid konfiguriran je za startanje neprekidne ADC pretvorbe dvije grupe ADC kanala. To je postignuto konfiguriranjem EOC (End Of Conversion) događaja koji označava kraj pretvorbe i istovremeno postavlja ADC prekid za ponovno startanje pretvorbe. Takva vrsta pretvorbe se na engleskom jeziku zove zove round robin pretvorba. ADC prekid koji je nazvan ADCinterupt1 bira koji će SOC okidač započeti ADC pretvorbu. ADCinterupt1 okida sve SOC-ove osim SOC3, SOC7, SOC11, SOC13 te će oni započeti ADC pretvorbu kao što je je prikazano na slici 4.30. [6]



Slika 4.30 Neprekidna ADC pretvorbena sekvenca [6]

Kako ADC-A3, ADC-A7, ADC-B3, ADC-B5, ADC-B7 analogni ulazi ne postoje na TMS320F28069 mikrokontroleru SOC3, SOC7, SOC11; SOC13 i SOC15 nisu korišteni u ovom programu. S obzirom da oni postoje mogli bi se konfigurirati da je to potrebno. Kanal ADC-B4 odnosno kraj njegove komunikacije EOC uzrokuje pokretanje ADCinterupt1 postupka. Prekid može biti stvoren prije nego što je ADC-B6 pretvoren zbog mogućnosti dvostrukog uzrokovanja ADC pretvarača. Pisanje u INPUTSEPOS polje ADCCTL1 kontrolnog registra omogućava ADCInterrupt1 da bude stvoren jedan ciklus prije njegovog kraja, kako je prikazano programskim kodom na slici 4.31.

```
// Stvori prekidni signal 1 ciklus prije njegovog kraja
AdcRegs.ADCCTL1.bit.INTPULSEPOS = 1;
AdcRegs.INTSEL1N2.bit.INT1SEL = 12; // ADCCH12 EOC uzrokuje
ADCInterrupt 1
AdcRegs.INTSEL1N2.bit.INT1CONT = 1; // Postavi ADCInterrupt
1 na automatsko brisanje
AdcRegs.INTSEL1N2.bit.INT1E = 1 // Omogući ADCInterrupt 1
```

Slika 4.31 Stvaranje ADCinterupta1 ciklus prije kraja pretvorbe [6]

ADCInterrupt1 konfiguriran je da generira serije SOC događaja koristeći ADCINTSOCSELx registar. Svaki od tih ADC SOC događaja konfiguriran je za startanje pretvorbe određenog ADC kanala sa CHSL (Channel Selection) područjem ADCSOCxCTL registra. CHSL područje se popunjava brojevima između 0 i 15 gdje analogni kanali ADCA0 do ADCA7 odgovaraju brojevima od 0 do 7, dok analogni kanali ADCB1 do ADCB7 odgovaraju brojevima od 8 do 15. Ako je SOCy polje ADCINTSOCSELx registra postavljeno na nulu, neće biti generiran prekid koji starta SOC. Postavljanje brojeva 1 ili 2 u ova polja omogućit će ADCInterrupt1 ili ADCInterrupt2 da starta početak ADC SOCx okidača za startanje pretvorbe. U ta polja postavlja se broj 1 kada se ADCInterrup1 koristi za početak svakog SOC-a. Opisani postupak je prikazan je programskim kodom na slici 4.32. [6]

```
AdcRegs.ADCINTSOCSEL1.bit.SOC0 = 1; // ADCInterrupt 1 uzrokuje
SOC0
...
...
AdcRegs.ADCINTSOCSEL2.bit.SOC14 = 1;
AdcRegs.ADCSOC0CTL.bit.CHSEL = 0; //Pretvori ADCA0(CH0) kada je
SOC0 primljen
AdcRegs.ADCSOC1CTL.bit.CHSEL = 1; // Pretvori ADCA1(CH1) kada je
SOC1 primljen
AdcRegs.ADCSOC2CTL.bit.CHSEL = 2;
AdcRegs.ADCSOC4CTL.bit.CHSEL = 4;
AdcRegs.ADCSOC5CTL.bit.CHSEL = 5;
AdcRegs.ADCSOC6CTL.bit.CHSEL = 6;
AdcRegs.ADCSOC8CTL.bit.CHSEL = 8; //Pretvori ADCB0(CH8) kada
SOC8 primljen
AdcRegs.ADCSOC9CTL.bit.CHSEL = 9; //Pretvori ADCB1(CH9) kada
SOC9 primljen
AdcRegs.ADCSOC10CTL.bit.CHSEL = 10;
AdcRegs.ADCSOC12CTL.bit.CHSEL = 12;
AdcRegs.ADCSOC14CTL.bit.CHSEL = 14;
```

Slika 4.32 Programski kod za startanje kontinuirane ADC pretvorbe [6]

Svaki ADCSOCxCTL registar ima 6-bitno područje ACQPS, koje određuje veličinu prozora za sampliranje. Vrijednost napisana u to područje za jedan je manja od željenog broja ciklusa sistemskog sata. Minimalan broj ciklusa sampliranja je 7, što znači da je ACQPS=6 te je tako postavljeno u ovom programu.

Nakon učitavanja programa u mikrokontroler, u prozor za promatranje i namještanje varijabli upisuje se AdcResults. Taj red će poslužiti za gledanje rezultata ADC pretvorbe svih jedanaest kanala, koliko ima na TMS320F28069 mikrokontroleru. Ako na neki kanal dovedemo sinusni napon, naša pretvorba odvijati će se na način da će

se rezultati pretvorbe tog kanala neprekidno mijenjati. Raspon brojeva pretvorenog kanala kretati će se od 0 do 4095 u decimalnom zapisu što je ekvivalento naponu od 0V do 3.3V.

4.9. PWM okidna ADC pretvorba

4.9.1. Svrha programa

Ovaj program demonstrira ADC pretvorbu gdje CMPA usporedbeni događaj EPWM vanjske jedinice započinje start pretvorbe u uzlaznom brojačkom modu. Sklopovska postavka mikrokontrolera radi se tako da se izlazni analogni signal koji stvara PWM1A dovede na ulaz ADC-A2 konvertera. [6]

Tablica 4.10 Prikaz sklopovskog povezivanja mikrokontrolera

1 ADC-A6 COMP3(+VE)	2 ADC-A2 COMP1 (+VE)	3 ADC-A0	4 3V3
5 ADC-A4 COMP2 (+VE)	6 ADC-B1	7 EPWM-4B GPIO-07	8 TZ1 GPIO-12
9 SCLA GPIO-33	10 ADC-B6 COMP3(-VE)	11 EPWM-4A GPIO-06	12 ADC-A1
13 SDAA GPIO-32	14 ADC-B0	15 EPWM-3B GPIO-05	16 5V0 (DisabledbyDefault)
17 EPWM-1A GPIO-00	18 ADC-B4 COMP2 (-VE)	19 EPWM-3A GPIO-04	20 SPISOMIA GPIO-17
21 EPWM-1B GPIO-01	22 ADC-A5	23 EPWM-2B GPIO-03	24 SPISIMOA GPIO-16
25 SPISTEA GPIO-19	26 ADC-B2 COMP1 (-VE)	27 EPWM-2A GPIO-02	28 GND
29 SPICLKA GPIO-18	30 GPIO-34 (LED)	31 PWM1A-DAC (Filtered)	32 GND

Nema spoja	Spojna žica

Nakon učitanja programa u mikrokontroler, sljedeće varijable dodaju se u prozor za njihovo praćenje:

- Duty_cycle_A (Format decimalni). Ova varijabla namješta vrijeme uključenosti PWM signala te se njenim mijenjanjem dobija različita vrijednost napona na izlazu.

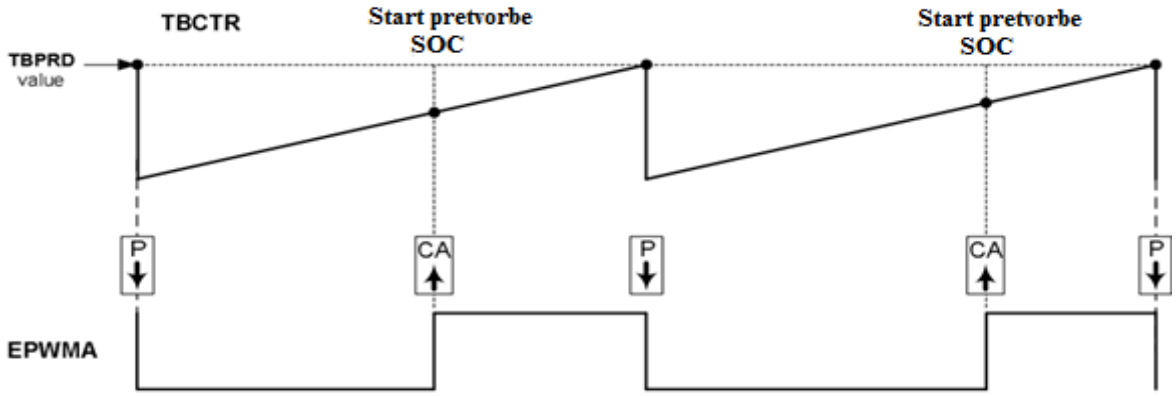
- Voltage_ADC2 (Format hexadecimalni). Ova varijabla pohranjuje rezultate ADC pretvorbe u rasponu 0x0000 do 0x0FFF, što predstavlja vrijednosti od 0V do 3.3V.

4.9.2. Opis rada programa

Za pokretanje pretvorbe odnosno SOC-a postoje različiti okidni mehanizmi koji mogu biti softverski, CPU prekidni tajmeri, XINT2 te EPWM signali. Ovim programom PWM1A je konfiguriran da pokrene ADC pretvorbu. Svaki put kada je usporedbeni A događaj jednak PWM1A brojaču koji radi u uzlaznom modu, ADC-A2 starta pretvorbu. Oblik generiranog pulsno širinskog valnog oblika određen je načinom na koji su određene postavke akcijskog kvalifikacijskog modula (Action Qualifier Module) konfigurirane te tako utječu na određene događaje brojača. Korištene su sljedeće postavke:

- Obriši PWM1A na događaj TBCTR=0 (Z)
- Postavi PWM1A na događaj TBCTR=CMPA [6]

Da bi start ADC pretvorbe (SOC) počeo u trenutku kad se dogodi usporedbeni događaj A, u TRIGSEL područje ADCSOC0CTL registra upisuje se broj 5. To omogućava start ADC pretvorbe u trenutku stvaranja PWM signala. Princip stvaranja signala i početak pretvorbe prikazan je slikom 4.33, dok je programsko konfiguriranje ADC-a prikazano slikom 4.34.

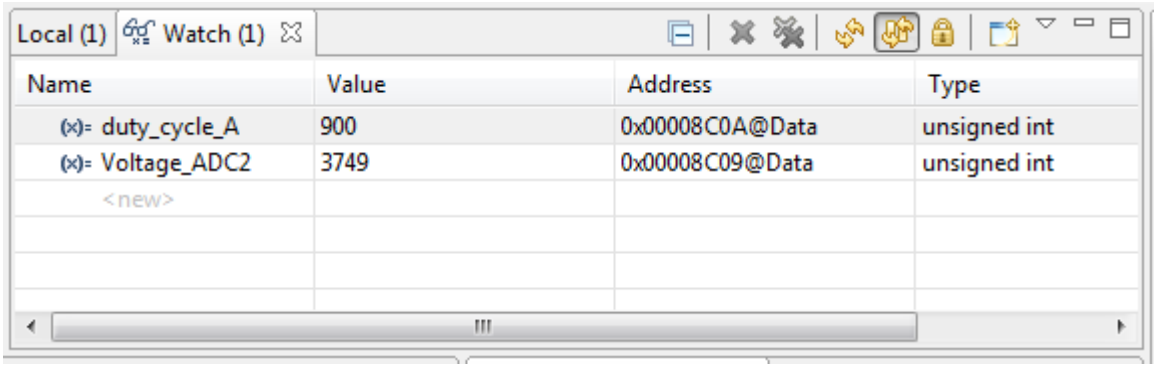


Slika 4.33 Stvaranje PWM signala i mjesto početka pretvorbe [6]

```
// Konfiguriranje ADC
AdcRegs.ADCSOC0CTL.bit.CHSEL    = 2;  //postavi SOC0 na
ADCINA2 kanal
AdcRegs.ADCSOC0CTL.bit.TRIGSEL  = 5;  //postavi SOC0 startni
okidač na  EPWM1A, u round-robin modu, SOC0 pretvara iza SOC1
AdcRegs.ADCSOC0CTL.bit.ACQPS    = 8;  //postavi SOC0 S/H
prozor na  8 ADC ciklusa sata
EDIS;
```

Slika 4.34 Konfiguriranje ADC okidača u Round robin modu [6]

S obzirom da se može namještati veličina prozora sampliranja, u ovom programu namještena je na 9 ciklusa sistemskog sata. Nakon učitanja programa u mikrokontroler, u prozor za promatranje varijabli postavlja se varijabla Voltage_ADC2 da bi se mogli promatrati rezultati ADC pretvorbe. Program sa pokreće u realnom vremenu, kako bi se mijenjanjem širine impulsa vidjele trenutačne digitalne vrijednosti pretvorbe napona.



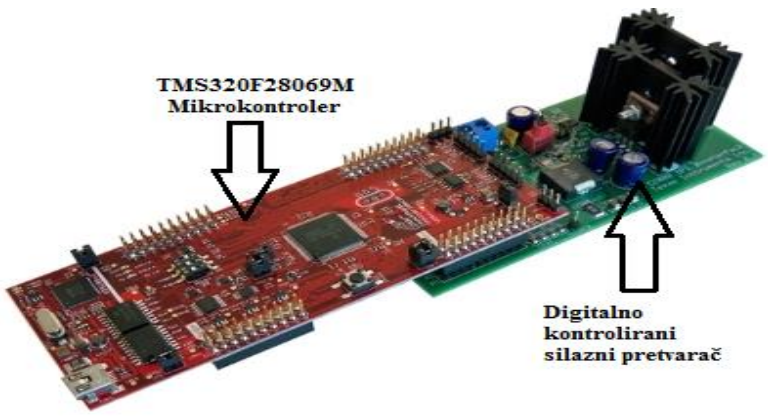
Slika 4.35 Prikaz pretvorene digitalne vrijednosti za određeno vrijeme vođenja PWM signala

5. SKLOPOVSKO POVEZIVANJE MIKROKONTROLERA TMS320F28069 U RAZLIČITIM SUSTAVIMA UPRAVLJANJA

5.1. Opis upravljanja istosmjernim pretvaračem korištenjem TMS320F28069M mikrokontrolera

Tradicionalni DC-DC pretvarači, odnosno istosmjerni upravljivi prekidački izvori napajanja bazirali su se na analognom upravljanju energetskim komponentama. Analogna tehnika bila je relativno jednostavna za implementaciju te je omogućavala relativno brzo upravljanje. Prvi digitalni sklopovi i digitalna računala bili su relativno spori pa se nisu mogli suprostaviti analognoj dominaciji, međutim razvoj mikrokontrolera u zadnjih dvadesetak godina omogućuje njihovu primjenu za upravljanje istosmjernim pretvaračima. Zbog njihove izrazito male potrošnje, velike brzine te mogućnosti rada u realnom vremenu porastao je interes u digitalno upravljano napajanje. S obzirom na veliku brzinu obrade podataka i mogućnost rada u zatvorenoj petlji, novi integrirani krugovi omogućili su bolje optimiranje algoritama upravljanja energetskim pretvaračima.

U daljnjem razmatranju pokazat će se način funkcioniranja neizoliranog DC-DC silaznog pretvarača, proizvedenog od tvrtke Texas Instruments, pod nazivom Digital Power BoosterPack. Navedenim pretvaračem može se upravljati korištenjem TMS320F28069M mikrokontrolera, kao što je prikazano na slici 5.1. Ova kombinacija pretvarača upravljanog mikrokontrolerom visokih performansi omogućuje široku primjenu prikazanog napajanja za pametne mreže, neprekidne izvore napajanja, industrijska napajanja, serverska napajanja, u autoindustriji te u ostalim suvremenim trošilima gdje su potrebna napajanja visoke učinkovitosti.

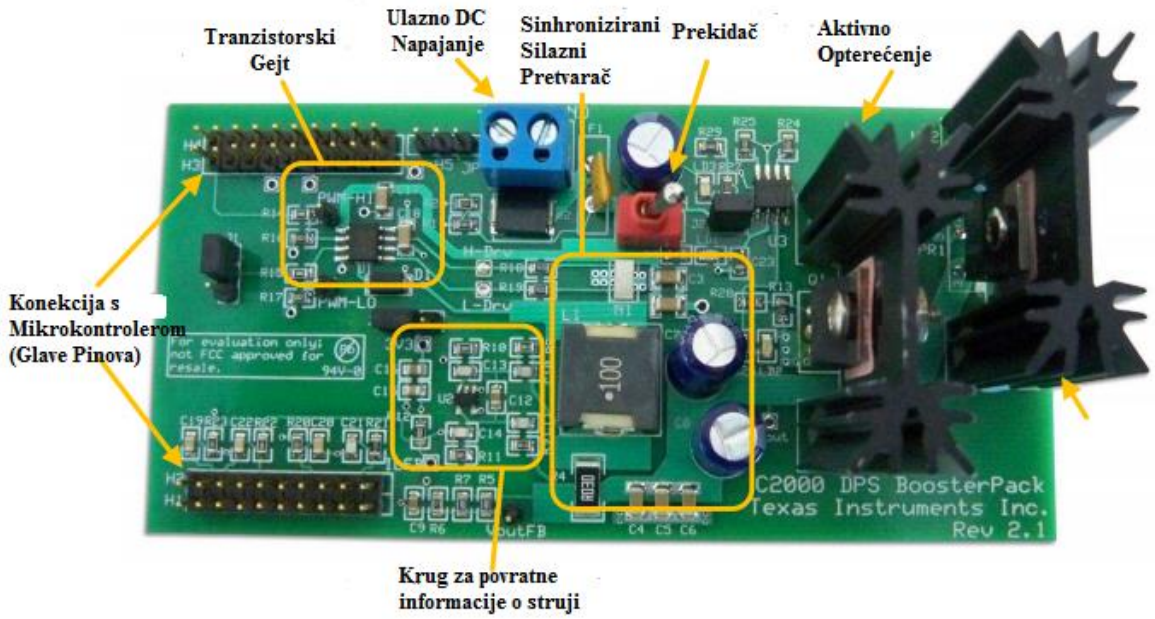


Slika 5.1 Digitalni silazni pretvarač

Korištenje prikazanog paketa mikrokontrolera visokih performansi i suvremenog pretvarača omogućuje brz i lagan način učenja o digitalnoj kontroli napajanja.

5.2. Glavne komponente silaznog digitalnog pretvarača

Slika 5.2. prikazuje glavne komponente silaznog digitalnog DC-DC pretvarača. TMS320F28069M mikrokontroler se priključuje na vanjsku periferiju pinova DC-DC pretvarača. Mikrokontroler upravlja izlaznim naponom silaznog digitalnog pretvarača. Pinovi mikrokontrolera upravljaju MOFSET (Metal Oxide Semiconductor Field Efect Transistor) tranzistorima dovodeći PWM signal na njihove upravljačke elektrode. Nominalno ulazno napajanje je 9V (minimalno 8.5V, maksimalno 12 V) te se može uključiti ili isključiti preko prekidača . Sinhronizirani silazni DC-DC pretvarač omogućuje regulaciju nominalnog izlaznog napona na 2V. Active load opterećenje ugrađeno na izlazu pretvarača, kojim se može softverski upravljati, pomaže ispitivanje performansi te ugađanju povratne veze. Krug za mjerenje struje omogućuje nadstrujnu zaštitu korištenjem povratne veze struje..



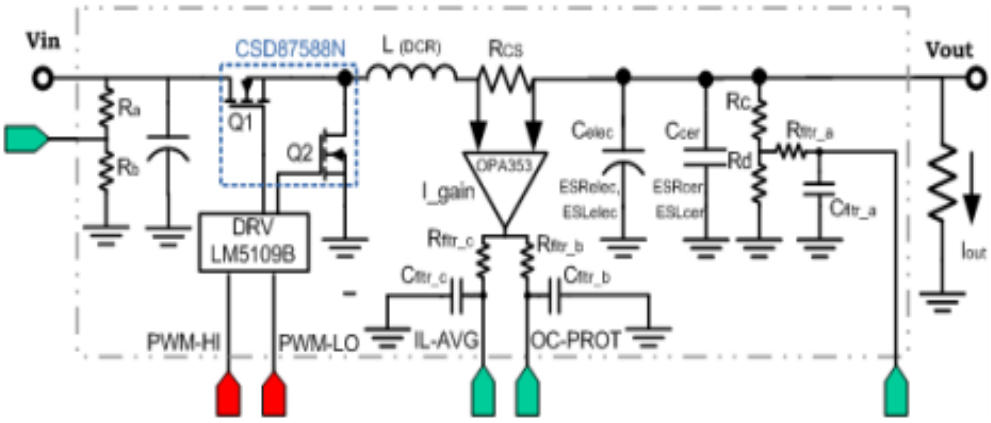
Slika 5.2 Glavne komponente silaznog digitalnog pretvarača [7]

5.3. Upravljački signali mikrokontrolera

Da bi mikrokontroler uspješno upravljao pretvaračem, mora istovremeno slati i primati određedene informacije. Signale koje mikrokontroler koristi prilikom upravljanja energetskim pretvaračem su:

- EPWM-4A Ovaj PWM signal upravlja MOFSET tranzistorom na visokoj strani sinhroniziranog pretvarača (PWM-HI), spojen na GPIO-6 pin mikrokontrolera.
- EPWM-4B Ovaj PWM signal upravlja MOFSET tranzistorom na nisokoj strani sinhroniziranog pretvarača (PWM-LO), spojen na GPIO-7 pin mikrokontrolera.
- EPWM-5B Alternativni upravljački PWM signal za nisku stranu (PWM-LO), spojen na GPIO-8 pin mikrokontrolera.
- EPWM-6A Kontrolni PWM signal za aktivno opterećenje, spojen na GPIO-10 pin mikrokontrolera.
- VoutFB-1 Povratna veza izlaznog napona, spojena na ADC-A3 pin mikrokontrolera.
- ILFB Strujna povratna veza (nadstrujna zaštita), spojena na ADC-A4 pin mikrokontrolera.
- ILFB_AVG Visoko filtrirana strujna povratna veza, spojena na ADC-B4 pin mikrokontrolera.
- VinFB Povratna veza ulaznog napona, spojena na ADC-B3 pin mikrokontrolera. [7]

Na slici 5.3 se vidi prikaz pojednostavljene strukture pretvarača, dok se kompletna električna shema nalazi u prilogu. Zelene strelice su signali povratne veze, dok su one označene crvenom bojom upravljački PWM signali MOFSET tranzistora. [7]

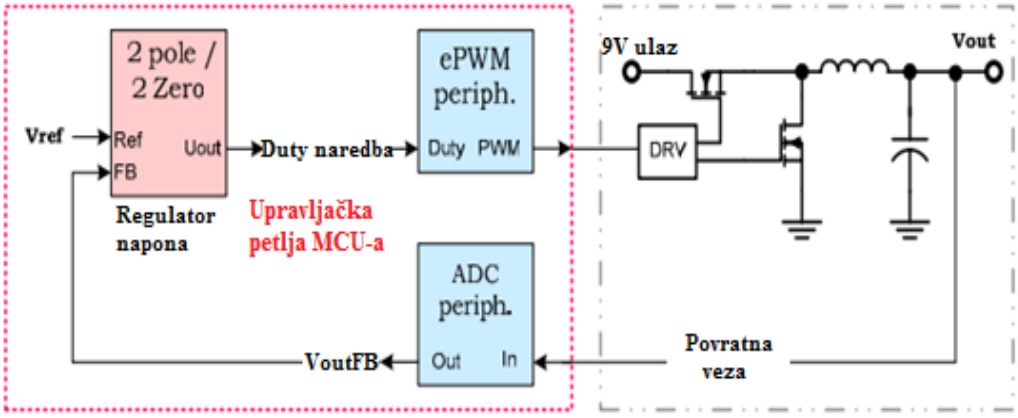


Slika 5.3 Pojednostavljena struktura silaznog pretvarača [7]

5.4. Upravljanje mikrokontrolerom

Da bi se digitalno upravljalo silaznim DC-DC pretvaračem, mora se upravljati vremenom vođenja dviju tranzistorskih MOFSET sklopki u ovisnosti o zadanim i

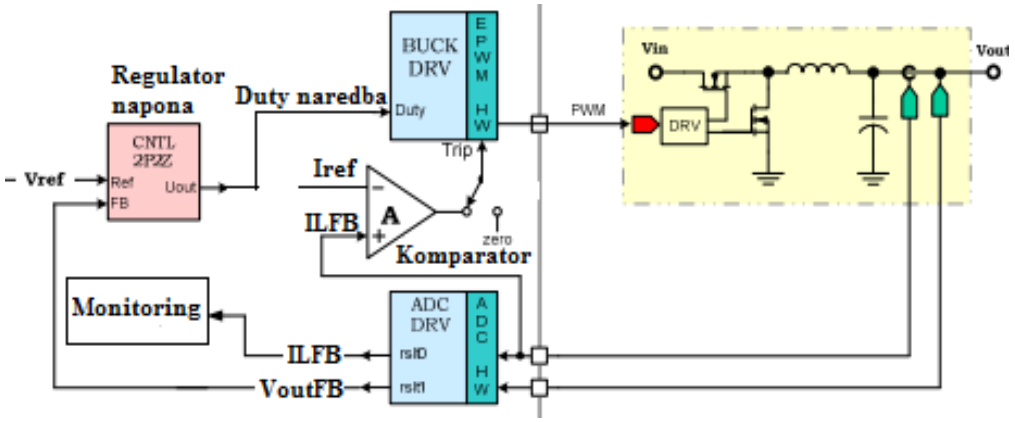
povratnim informacijama poput izlazne struje i napona. Mikrokontroler TMS320F28069M prati i obrađuje navedene signale te na temelju njih stvara dva PWM signala frekvencije 200 kHz. Ta dva signala (EPWM-4A i EPWM-4B) dovode se na gate driver, što je pojačalo snage dovedenog signala. Ono diže signale na viši naponski nivo, da bi se stvorila dovoljno velika struja za aktiviranje upravljačke elektrode MOFSET tranzistora, odnosno da bi tranzistori proveli. Što je vrijeme vođenja tranzistora veće, biti će veća naponska razina na izlazu. Na slici 5.4. vidi se pojednostavljeni proces upravljanja MCU-a temeljen na zadanom referentnom izlaznom naponu.



Slika 5.4 Naponski mod upravljanja pretvaračem

U naponskom modu TMS320F28069M mikrokontroler prima povratnu informaciju o trenutnom izlaznom analognom naponu (Vout) te ga nakon pretvorbe u digitalni (VoutFB) uspoređuje sa zadanim referentnim (Vref) naponom. Tu uspredbu radi kontrolni blok regulator napona, što je u biti PID (Proporcionalno integralno derivacijski) regulator. Reglator drugog reda je softverski napravljen te ima dva pola i dvije nule, a može se koristiti i regulator višeg reda. Njegov izlaz daje naredbu za promjenu širine impulsa (Duty) PWM jedinici za pulsno širinsku modulaciju koja se nalazi u mikrokontroleru. Ovakvim upravljanjem se realizira softverska regulacija koja daje stabilan napon na izlazu DC-DC pretvarača. Analogni komparator na TMS320F28069 mikrokontroleru te DAC mehanizam zadužen za pretvaranje zadane digitalne referentne vrijednosti u analognu, omogućuje nadstrujnu zaštitu sustava. Na invertirajući ulaz komparatora dovodi se referentna, odnosno zadana vrijednost DAC struje. DAC struja predstavlja gornju granicu izlazne struje koju sustav može podnijeti. Referentna vrijednost struje uspoređuje se s analognom ILFB povratnom vezom trenutne izlazne struje iz DC-DC pretvarača. Onog trenutka kad izlazna struja bude veća od zadane referentne struje komparator će aktivirati okidni trip događaj PWM vanjskoj jedinici .

Okidni trip događaj će automatski zaustaviti generaciju EPWM-4A i EPWM-4B signala kako bi se zaštito DC-DC pretvarač od pregaranja. Opisani način rada prikazan je slikom 5.5.



Slika 5.5 Nadstrujna zaštita pretvarača u naponskom modu

6. ZAKLJUČAK

Kako se vidi iz prikazanih programa, TMS320F28069 mikrokontroler nudi široku lepezu mogućnosti za ranolike sustave upravljanja baziranih na principu zatvorene petlje, kao što su kontrola elektromotora, upravljanje pretvaračima u sustavima za dobivanje solarne energije, upravljanje naponskim napajanja, auto industriji te još mnogo drugih sustava. Iz arhitekture mikrokontrolera TMS320F28069 vidi se da je on dominantno projektiran za digitalno upravljanje pretvaračima napona i elektromotornim pogonima. Mnoštvo konfiguracijskih opcija omogućuje stvaranje poboljšanih PWM signala koji mogu biti konfigurirani prema nuli, ukupnom periodu te tajmer događajima CMPA i CMPB. Neovisno namještanje padjućih i rastućih bridova PWM signala sprječava nepotrebne gubitke energije. S obzirom da PWM vanjske jedinice podržavaju fazna odsupanja te sinkronizaciju signala, otvara se mogućnost upravljanja raznim vrstama DC-DC pretvarača. Korištenje jednog HRPWM-a na TMS320F20069 mikrokontroleru pruža mnogo finiju kontrolu na frekvencijama većim od 200 kHz. Kako je rezolucija HRPWM-a i do 100 puta veća od konvencionalnih PWM modula, postiže se puno preciznija kontrola na rastućim i padajućim bridovima PWM signala. Integrirani analogni komparatori omogućuju nadstrujnu i nadnaponsku zaštitu sustava, kao i zaštitu od vršnih struja aktivirajući trip okidne događaje. Visoko brzinski ADC pretvarač sa dvostrukim simultanim sampliranjem omogućuje vrlo brzu pretvorbu 11 analognih signala dovedenih s određenih osjetila koji se nalaze u procesu. Mogućnost rada u realnom vremenu jedna je od najvažnijih značajki mikrokontrolera. Za razliku od starijih mikrokontrolera koji nisu bili u stanju vrlo brzo izračunati iznose upravljačkih signala prilikom korištenja naprednih metoda upravljanja, TMS320F28069 mirokontroler omogućava proračun složenih upravljačkih signala između dva perioda uzrokovanja analognog signala.

Na kraju rada prikazan je i analiziran primjer digitalnog upravljanja istosmjernim silaznim pretvaračem. Iz opisanog primjera, vidljivo je da osim osnovnih funkcija upravljanja, mikrokontroler TMS320F28069 omogućava realizaciju zaštitnih funkcija pretvarača.

U Varaždinu, 19.07.2016.

Hrvoje Čikeš.

Sveučilište
Sjever

IZJAVA O AUTORSTVU I SUGLASNOST ZA JAVNU OBJAVU

Ja, Hrvoje Čikeš pod punom moralnom, materijalnom I kaznenom odgovornošću, izjavljujem da sam isključivi autor završnog rada pod naslovom *Analiza mogućnosti upotrebe mikrokontroleraTMS320F28069 u sustavima upravljanja* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student: H. Čikeš
(vlastoručni potpis)

Ja, Hrvoje Čikeš neopozivo izjavljujem da sam suglasan s javnom objavom završnog rada pod naslovom *Analiza mogućnosti upotrebe mikrokontroleraTMS320F28069 u sustavima upravljanja* čiji sam autor.

Student: H. Čikeš
(vlastoručni potpis)

7. LITERATURA

[1] TMS320x2806x Piccolo Technical Reference Guide (Rev.F):
http://www.ti.com/lit/ug/spruh18f/spruh18f.pdf , dostupno dana 08.06.2016.
[2] Datasheet TMS320F2806x Piccolo Microcontrollers (Rev.F):
http://www.ti.com/lit/ds/symlink/tms320f28069.pdf, dostupno dana 06.06.2016.
[3] TMS320C28x Extended Instruction Sets Tehnical Reference Manual (Rev.A)
http://www.ti.com/lit/ug/spruhs1a/spruhs1a.pdf, dostupno dana 20.05.2016.
[4] C2000 Microcontrollers Brochure (Rev.AC):
http://www.ti.com/lit/sg/sprb176ac/sprb176ac.pdf, dostupno dana 16.05.2016.
[5] C2000 One-Day Workshop:
http://processors.wiki.ti.com/index.php/C2000_One-Day_Workshop#Workshop_Materials_-_Revision_4.2 , dostupno dana 18.05.2016.
[6] Control Suite Example Projects Piccolo TMS320F28069 ControlStick:
http://www.ti.com/tool/controlsuite, dostupno dana 08.06.2016.
[7] C2000™ Digital Power BoosterPack:
http://www.ti.com/lit/ug/tidu986/tidu986.pdf, dostupno dana 20.05.2016.

8. PRILOZI

8.1. Programski kod za simetričnu pulsno širinsku modulaciju [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"          //  Korisne  definicije
za  inicijalizaciju

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash();
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Općenita deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;

Uint16 duty_cycle_A=250;   // Postavi duty na 50%
Uint16 duty_cycle_B=250;   // Postavi duty na 50%

// Početak glavnog koda

void main(void)
{
// Generalna inicijalizacija

     DeviceInit();   //   Podrška   mikrokontrolera   i   GPIO
multipleksera

// Koristi se samo ako se pokreće iz flash memorije
#ifdef FLASH
// Kopiraj kritičan kod i FLASH postavke u RAM
// RamfuncsLoadStart, RamfuncsLoadEnd i RamfuncsRunStart
// Simboli su kreirani povezivačkim programom.
     MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);
//  Pozovi  FLASH  inicijalizaciju  za  namještanje  FLASH  stanja
čekanja
     InitFlash();    // Pozovi flash zaštitnu  inicijalizacijsku
funkciju
#endif //(FLASH)

#define period 500    // 80kHz kad je PLL postavljen 0x10 (80MHz)
//      period 250   // 160kHz kad je PLL postavljen 0x10 (80Mhz)

  // Vremensko-bazni registri
```

```
      EPwm2Regs.TBPRD = period;              //   Postavi   period
tajmera, PWM frekvencija = 1 / period
      EPwm2Regs.TBPHS.all = 0;   // Vremensko bazni registar faze
      EPwm2Regs.TBCTR = 0;      //   Vremensko   bazni   brojački
registar
       EPwm2Regs.TBCTL.bit.PRDLD  =  TB_IMMEDIATE;    //  Postavi
trenutno dizanje
       EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN; // Uzlazno
silazni mod korišten za simetrične PWM signale
      EPwm2Regs.TBCTL.bit.PHSEN = TB_DISABLE;       //  Onemogući
dizanje faze
      EPwm2Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
      EPwm2Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
      EPwm2Regs.TBCTL.bit.CLKDIV = TB_DIV1;

                                              // Postavi shadow
register dizanje na nulu

      EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
      EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
      EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;    //  Dizanje
na CTR=Zero
      EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;    //  Dizanje
na CTR=Zero

      // Postavi usporedbene vrijednosti

      EPwm2Regs.CMPA.half.CMPA = duty_cycle_A;    // Postavi duty
na 50% u početku
      EPwm2Regs.CMPB = duty_cycle_B;              // Postavi duty
na 50% u početku

      // Postavljanje akcija akcijskog kvalifikacijskog modula

      EPwm2Regs.AQCTLA.bit.CAU = AQ_SET;               // Postavi
PWM2A na događaj A uzlazno brojanje
      EPwm2Regs.AQCTLA.bit.CAD = AQ_CLEAR;             // Obriši
PWM2A na događaj A silazno brojanje
      EPwm2Regs.AQCTLB.bit.CBU = AQ_SET;               // Postavi
PWM2B na događaj B, uzlazno brojanje
      EPwm2Regs.AQCTLB.bit.CBD = AQ_CLEAR;         // Obriši PWM2B
na događaj B, silazno brojanje

//Beskonačna petlja

       for(;;)

      {
      EPwm2Regs.CMPA.half.CMPA   =   duty_cycle_A;      //   Dodaj
duty_cycle_A u watch window
      EPwm2Regs.CMPB = duty_cycle_B;             //          Dodaj
duty_cycle_B u watch window
      }

}
```

8.2. Programski kod za asimetričnu pulsno širinsku modulaciju [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"          //  Korisne  definicije
za inicijalizaciju

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash(void);
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Deklaracija varijabli - Općenita

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;
Uint16 duty_cycle_A=500;   // postavi duty  na 50% u početku
Uint16 duty_cycle_B=500;   // postavi duty  na 50% u početku

// Glavni kod

void main(void)
{
//    Općenita inicijalizacija

      DeviceInit();

// Koristi se samo ako se pokreće iz FLASH memorije
#ifdef FLASH
      MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);

      InitFlash();
#endif //(FLASH)
#define period 1000   // 80kHz kad je PLL postavljen 0x10 (80MHz)
  //    period 500    // 160kHz when je PLL postavljen 0x10(80MHz)

// Vremensko-bazni registri

      EPwm2Regs.TBPRD = period;          //     Postavi     period
tajmera, PWM frequency = 1 / period
      EPwm2Regs.TBPHS.all = 0;   // Vremensko bazni fazni registar
      EPwm2Regs.TBCTR = 0;       //   Vremensko   bazni   brojački
registar
     EPwm2Regs.TBCTL.bit.PRDLD   =   TB_IMMEDIATE;     //   Postavi
automatsko dizanje
     EPwm2Regs.TBCTL.bit.CTRMODE  =  TB_COUNT_UP; // Uzlazni mod
korišten za simetrični PWM
      EPwm2Regs.TBCTL.bit.PHSEN = TB_DISABLE;   // Onemogući fazno
dizanje
      EPwm2Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
```

```
      EPwm2Regs.TBCTL.bit.CLKDIV = TB_DIV1;

      // Postavi dizanje shadow registra na nulu

      EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
      EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
      EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;// Dizanje na
CTR=Zero
      EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;// Dizanje na
CTR=Zero

      // Postavi usporedbene vrijednosti

      EPwm2Regs.CMPA.half.CMPA = duty_cycle_A; // Postavi duty u
početku na 50%
      EPwm2Regs.CMPB = duty_cycle_B;           // Postavi duty u
početku na 50%

     // Postavljanje akcija

      EPwm2Regs.AQCTLA.bit.ZRO = AQ_SET;     // Postavi PWM2A na
nulu
      EPwm2Regs.AQCTLA.bit.CAU = AQ_CLEAR;   // Izbriši PWM2A na
događaj A, uzlazno brojanje

      EPwm2Regs.AQCTLB.bit.ZRO = AQ_CLEAR;   // Postavi PWM2B na
nulu
      EPwm2Regs.AQCTLB.bit.CBU = AQ_SET;     // Obriši PWM2B na
događaj B , uzlazno brojanje

      //    Beskonačna petlja

       for(;;)
      {
       EPwm2Regs.CMPA.half.CMPA = duty_cycle_A;         // Dodaj
duty_cycle_A u watch window
       EPwm2Regs.CMPB = duty_cycle_B; // Dodaj duty_cycle_B u
watch window i promijeni vrijednost da vidiš efekt
      }

}
```

8.3. Programski kod za asimetrični PWM sa mrtvim vremenom (DeadBand PWM) [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"

void DeviceInit(void);
void InitFlash(void);
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);
```

```
// Općenita deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;

Uint16  RED_delay=30;  //  375ns  kada  je  PLL  postaljen  na  0x10
(80MHz)
Uint16  FED_delay=30;  //  375ns  kada  je  PLL  postaljen  na  0x10
(80MHz)

// Početak glavnog koda

void main(void)
{

//    Općenita inicijalizacija

      DeviceInit();
#ifdef FLASH

      MemCopy(&RamfuncsLoadStart,                &RamfuncsLoadEnd,
&RamfuncsRunStart);

      InitFlash();
#endif //(FLASH)


#define period 1000   // 80kHz kada je PLL postavljen na 0x10
(80MHz)
  //     period 500   // 160kHz kada je PLL postavljen na 0x10
(80MHz)

  // Vremensko bazni registri

      EPwm2Regs.TBPRD = period;                 // Postavi period
tajmera, PWM frequency = 1 / period
      EPwm2Regs.TBPHS.all = 0;   // Vremensko bazni fazni registar
      EPwm2Regs.TBCTR = 0;       //   Vremensko   bazni   brojački
registar
       EPwm2Regs.TBCTL.bit.PRDLD  =  TB_IMMEDIATE;     //  Postavi
automatsko dizanje
     EPwm2Regs.TBCTL.bit.CTRMODE   =   TB_COUNT_UP;   //   Uzlazni
brojački mod za asimetrični PWM
      EPwm2Regs.TBCTL.bit.PHSEN = TB_DISABLE;   // Onemogući fazno
dizanje
      EPwm2Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
      EPwm2Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
      EPwm2Regs.TBCTL.bit.CLKDIV = TB_DIV1;

      // Postavi dizanje shadow registra na nulu
      EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
      EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
      EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
      EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;
```

```
      // Postavi usporedbene događaje
      EPwm2Regs.CMPA.half.CMPA  =  period/2;  //  Duty  je  50%  na
početku
      EPwm2Regs.CMPB = period/2;           //  Duty  je  50%  na
početku

      // Postavi akcije

      EPwm2Regs.AQCTLA.bit.ZRO = AQ_SET;      // Postavi PWM2A na
nulu
      EPwm2Regs.AQCTLA.bit.CAU = AQ_CLEAR;    // Obriši PWM2A na
događaj A, uzlazno brojanje

      EPwm2Regs.AQCTLB.bit.ZRO = AQ_CLEAR;    // Postavi PWM2B na
nulu
      EPwm2Regs.AQCTLB.bit.CBU = AQ_SET;      // Obriši PWM2B na
događaj B, uzlazno brojanje

    //  Konfiguracija DeadBand podmodula

      EPwm2Regs.DBCTL.bit.IN_MODE = DBA_ALL;          //   EPWM2A
je izvor za kašnjenje padajućeg i rastućeg brida
      EPwm2Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE;  // FED i RED
su pridruženi ulaznim signalima
      EPwm2Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;       //   Active
High Complementary (AHC)
      EPwm2Regs.DBRED = RED_delay;        //    Postavi    vrijeme
kašnjenja za rastući brid
      EPwm2Regs.DBFED = FED_delay;        //    Postavi    vrijeme
kašnjenja za padajući brid

  // Beskonačna petlja

      for(;;)
      {
            EPwm2Regs.DBRED = RED_delay;     // Promijeni kašnjenje
rastućeg brida iz watch window
            EPwm2Regs.DBFED = FED_delay;     // Promijeni kašnjenje
rastućeg brida iz watch window
      }

}
```

8.4. Programski kod za okidni PWM [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash(void);
```

```
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Generalna deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;
Uint16 one_shot=1;              // Postavi kao 0 za CBC
                                // Postavi kao 1 za OST
// Početak glavnog koda

void main(void)
{

// Generalna inicijalizacija
     DeviceInit();
)
#ifdef FLASH
     MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);

     InitFlash();    #endif //(FLASH)

// PWM1 i TZ1 Konfiguracija

#define period 100 // 800kHz kada je PLL postavljen na 0x10 (80MHz)
//     period 50 // 1600kHz kada je PLL postavljen na 0x10 (80MHz)

   EALLOW;

   if(one_shot==1)
   EPwm1Regs.TZSEL.bit.OSHT1 = 1;          //  Konfiguriraj  TZ1
kao one shot trip
   else
   EPwm1Regs.TZSEL.bit.CBC1 = 1;      // Konfiguriraj TZ1 kao CBC

//Što zelimo da TZ1 radi
   EPwm1Regs.TZCTL.bit.TZA = TZ_FORCE_HI;
   EPwm1Regs.TZCTL.bit.TZB = TZ_FORCE_LO;

   EDIS;

  // Vremensko bazni registri

     EPwm1Regs.TBPRD = period;   // Postavi period tajmera, PWM
frequency = 1 / period
     EPwm1Regs.TBPHS.all = 0;    // Vremensko bazni registar
     EPwm1Regs.TBCTR = 0;        //   Vremensko   bazni   brojački
registar
     EPwm1Regs.TBCTL.bit.PRDLD   =   TB_IMMEDIATE;     //   Postavi
automatsko dizanje
     EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP; // Uzlazni mod za
asimetrični PWM
     EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;       //  Onemogući
fazno dizanje
     EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
```

```
      EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
      EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;

 // Postavi dizanje shadow registra na nulu
      EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
      EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
      EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
      EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;

 // Postavi usporedbene vrijednosti

      EPwm1Regs.CMPA.half.CMPA = period/2;    // Postavi duty na
50% u početku
      EPwm1Regs.CMPB = period/2;     //  Postavi  duty  na  50%  u
početku
      EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;      // Postavi PWM2A na
nulu
      EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;    // Obriši PWM2A na
događaj A, uzlazno brojanje

      EPwm1Regs.AQCTLB.bit.ZRO = AQ_CLEAR;    // Postavi PWM2B na
nulu
      EPwm1Regs.AQCTLB.bit.CBU = AQ_SET;      // Obriši PWM2B na
događaj B, uzlazno brojanje

//    Beskonačna petlja

      for(;;)
      {

      }


} // Kraj glavnog koda
```

8.5. Programski kod za filtrirani visokorezolucijski PWM [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash(void);
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Generalna deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;
Uint32 duty_cycle;
```

```
// Početak glavnog koda

void main(void)
{

// Općenita inicijalizacija

     DeviceInit();

#ifdef FLASH

     MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);

     InitFlash();
#endif //(FLASH)


// PWM inicijalizacija

#define period 40    // 2MHz kada je PLL postavljen na  0x10
(80MHz)


  // Vremensko bazni registri
     EPwm1Regs.TBPRD = period;       // Postavi period tajmera,
PWM frequency = 1 / period
     EPwm1Regs.TBPHS.all = 0;        // Vremensko bazni registar
     EPwm1Regs.TBCTR = 0;     //   Vremensko   bazni   brojački
registar
     EPwm1Regs.TBCTL.bit.PRDLD  =  TB_IMMEDIATE;     //  Postavi
automatsko dizanje
     EPwm1Regs.TBCTL.bit.CTRMODE   =   TB_COUNT_UP;   //   Uzlazni
brojački mod za asimetrični PWM
     EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;      //   Onemogući
fazno dizanje
     EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
     EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
     EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;

     // Postavi dizanje shadow registra na nulu
     EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;

     // Postavi usporedbene vrijednosti
     EPwm1Regs.CMPA.half.CMPA = 20;          //   Postavi   duty
početno na 50%

     // Postavi akcije
     EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;      // Postavi PWM1A na
početak brojanja
     EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;     // Obriši PWM1A na
događaj A uzlaznog brojanja

     // Postavi visokorezolucijske registre
```

```
      EALLOW;
      EPwm1Regs.HRCNFG.all = 0x0;
//Inicijaliziraj visokorezolucijski PWM kontrolni registar
      EPwm1Regs.HRCNFG.bit.EDGMODE   =   HR_REP;         //Omogući
visokorezolucijsku kontrolu na rastućem bridu
      EDIS;

//    Peskonačna petlja

      duty_cycle = 0x1400;        // Postavi duty_cycle varijablu
na 50%

      for(;;)  //Beskonačna petlja
      {
                  EPwm1Regs.CMPAM.all = duty_cycle<<8;  // Ažuriraj
usporedbeni  A registar koristeći uniju CMPA i CMPAHR  registra
// Ova vrijednost mijenja PWM1A_DAC izlaza

      }
} //Kraj glavnog koda
```

8.6. Programski kod za uspoređivanje analognih vrijednosti komparatorom [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash();
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Općenita deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;

Uint16 dac_value= 310;                // Postavi na 1V u početku
Uint16 duty_cycle_A=20;               // Postavi duty početno na
50% 1.65 V

// Glavni kod počinje ovdje

void main(void)
{
// Općenita inicijalizacija

DeviceInit();   s
#ifdef FLASH
      MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);
```

```
     InitFlash();
#endif //(FLASH)

   EALLOW;
   AdcRegs.ADCCTL1.bit.ADCBGPWD = 1;         //Koristi BG napon
   Comp1Regs.COMPCTL.bit.COMPDACEN = 1;      //Uključi komparator
// Comp1Regs.COMPCTL.bit.COMPSOURCE = 1;    //Spoji invertirajući
ulaz na pin COMP1B
//Odkomentiraj prethodnu liniju za korištenje COMP1B umjesto
unutarnjeg DAC napona

Comp1Regs.COMPCTL.bit.COMPSOURCE = 0;  //Spoji invertirajući ulaz
na unutarnji DAC
   Comp1Regs.DACVAL.bit.DACVAL =dac_value; //Postavi DAC izlaz na
srednju točku

// PWM-DAC & TZ konfiguracija

#define period 40  // 2MHz kada je  PLL postavljen na 0x10 (80MHz)


// Vremensko bazni registri

     EPwm1Regs.TBPRD = period;        // Postavi period tajmera,
PWM frequency = 1 / period
     EPwm1Regs.TBPHS.all = 0;   // Vremensko bazni fazni registar
     EPwm1Regs.TBCTR = 0;       //   Vremensko   bazni   brojački
registar
     EPwm1Regs.TBCTL.bit.PRDLD   =   TB_IMMEDIATE;      //Postavi
automatsko dizanje
     EPwm1Regs.TBCTL.bit.CTRMODE   =   TB_COUNT_UP;   //   Uzlazni
brojački mod za asimetrični PWM
     EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;       //Onemogući
fazno dizanje
     EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
     EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
     EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;

     // Postavi dizanje  shadow registra na nulu
     EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;

     // Postavi usporedbene vrijednosti
     EPwm1Regs.CMPA.half.CMPA = 20;   // Početno postavi duty na
50%

     EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;      // Postavi PWM2A na
nulu
     EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;    // Obriši PWM2A na
događaj A, uzlazno brojanje

     EALLOW;

// Konfiguriraj digitalno usporedbeni događaj (DCAEVT2)
```

```
    EPwm1Regs.DCTRIPSEL.bit.DCAHCOMPSEL = DC_COMP1OUT;    // DCAH
= Comparator 1 izlaz
    EPwm1Regs.DCTRIPSEL.bit.DCALCOMPSEL = DC_TZ2;         // DCAL
= TZ2
    EPwm1Regs.TZDCSEL.bit.DCAEVT2 = TZ_DCAH_LOW;             //
DCAEVT2 =  DCAH nisko(Postati će aktivno ako izlaz komparatora
ode nisko)
    EPwm1Regs.DCACTL.bit.EVT1SRCSEL = DC_EVT2;          // DCAEVT2
= DCAEVT2 (nije filtrirano)
    EPwm1Regs.DCACTL.bit.EVT1FRCSYNCSEL = DC_EVT_ASYNC;  // Take
async path


// Uključi DCAEVT2  DCBEVT2 CBC trip događaje

    EPwm1Regs.TZSEL.bit.DCAEVT2 = 1;
    EPwm1Regs.TZSEL.bit.DCBEVT2 = 1;
    EPwm1Regs.TZCTL.bit.TZA = TZ_NO_CHANGE;     // Onemogući TZ
akcije
    EPwm1Regs.TZCTL.bit.DCAEVT2 = TZ_NO_CHANGE;  // Onemogući
DCAEVT2 akcije
     EPwm1Regs.TZCLR.all = 0xFFFF;
     EDIS;

//   Beskonačna petlja

     for(;;)
     {

          if(EPwm1Regs.TZFLG.bit.DCAEVT2==0)     // Uključi LED
LD2 kada komparator ode nisko

             GpioDataRegs.GPBSET.bit.GPIO34 = 1;
          else
             GpioDataRegs.GPBCLEAR.bit.GPIO34 = 1;


          EALLOW;
          EPwm1Regs.TZCLR.bit.DCAEVT2 =0xFFFF;// Obriši DCAEVT2
zastavicu
          EDIS;
     //

          Comp1Regs.DACVAL.bit.DACVAL =dac_value;        //
Namjesti unutarnju DAC vrijednost
          EPwm1Regs.CMPA.half.CMPA = duty_cycle_A;   // Namjesti
PWM1A-DAC izlaz
     }


   //Beskonačna petlja


}
```

8.7. Programski kod za okidni PWM generiran uporedbom analognih signala [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash(void);
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Općenita deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;
Uint16 one_shot=1;              // Postavi na 0 for Za CBC
                                      // Postavi na 1 za OST
```

```
// Početak glavnog koda

void main(void)
{

//Generalna inicijalizacija

     DeviceInit();
#ifdef FLASH
     MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);

     InitFlash();
#endif //(FLASH)

// PWM1 & TZ1 konfiguracija

#define period 100   // 800kHz kada je PLL postavljen na 0x10
(80MHz)
//       period 50 // 1600kHz kada je  PLL postavljen na 0x10
(80MHz)

   EALLOW;

   if(one_shot==1)
   EPwm1Regs.TZSEL.bit.OSHT1 = 1;     // Konfiguriraj TZ1 kao OST
   else
   EPwm1Regs.TZSEL.bit.CBC1 = 1;      // Konfiguriraj TZ1 kao CBC

   // Ono što želimo da TZ1 radi?
   EPwm1Regs.TZCTL.bit.TZA = TZ_FORCE_HI;
   EPwm1Regs.TZCTL.bit.TZB = TZ_FORCE_LO;

   EDIS;

  // Vremensko bazni registri

     EPwm1Regs.TBPRD = period;         // Postavi period tajmera,
PWM frequency = 1 / period
     EPwm1Regs.TBPHS.all = 0;   // Vremensko bazni registar
     EPwm1Regs.TBCTR = 0;       //   Vremensko   bazni   brojački
registar
     EPwm1Regs.TBCTL.bit.PRDLD   =   TB_IMMEDIATE;      //Postavi
automatsko dizanje
     EPwm1Regs.TBCTL.bit.CTRMODE   =   TB_COUNT_UP;   //   Uzlazni
brojački mod za asimetrični PWM
     EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;   // Onemogući fazno
dizanje
     EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
     EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
     EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;

     // Postavi dizanje shadow registra na nulu

     EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
     EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
```

```
      EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
      EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;

      // Postavi usporedbene vrijednosti

      EPwm1Regs.CMPA.half.CMPA = period/2; // Postavi duty na 50%
početno
      EPwm1Regs.CMPB = period/2;       //  Postavi  duty  na  50%
početno

     // Postavljanje akcija

      EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;      // Postavi PWM2A na
nulu
      EPwm1Regs.AQCTLA.bit.CAU  =  AQ_CLEAR;  //  Obriši  PWM2A  na
događaj A, uzlazno brojanje

      EPwm1Regs.AQCTLB.bit.ZRO  =  AQ_CLEAR;  //  Postavi  PWM2B  na
nulu
      EPwm1Regs.AQCTLB.bit.CBU  =  AQ_SET;     //  Obriši  PWM2B  na
događaj B, uzlazno brojanje


      for(;;)  //Beskonačna petlja
      {

      }



}
```

8.8. Programski kod za neprekidnu ADC pretvorbu [6]

```
#include "PeripheralHeaderIncludes.h"

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash(void);
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Općenita deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;

Uint16 AdcResults[16];

// Početak glavnog koda
```

```
void main(void)
{
      Uint16 i = 0;

// Općenita inicijalizacija

      DeviceInit();   //
#ifdef FLASH
      MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);

      InitFlash();
#endif //(FLASH)

///   Inicijalizacija vanjskih jedinica

// ADC Inicijalizacija
    EALLOW;
      AdcRegs.ADCCTL1.bit.ADCREFSEL   = 0; // Koristi unutarnje
bandgap napajanje
      AdcRegs.ADCCTL1.bit.ADCBGPWD    = 1; // Uključi band gap
      AdcRegs.ADCCTL1.bit.ADCREFPWD   = 1; // Uključi referencu
      AdcRegs.ADCCTL1.bit.ADCPWDN     = 1; //   Uključi   ostatak
ADC-a
      AdcRegs.ADCCTL1.bit.ADCENABLE   = 1; // Omogući ADC
    for(i=0; i<5000; i++){}                          //    Čekaj
60000 ciklusa = 1ms (svaka iteracija je 12 ciklusa)

      AdcRegs.ADCCTL1.bit.INTPULSEPOS = 1; //  Stvori  prekid  1
ciklus prije njegovog kraja

      // postavi S/H prozor na 6 ciklusa sata (112.5ns)
      AdcRegs.ADCSOC0CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC1CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC2CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC4CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC5CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC6CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC9CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC8CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC10CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC12CTL.bit.ACQPS = 8;
      AdcRegs.ADCSOC14CTL.bit.ACQPS = 8;


      AdcRegs.INTSEL1N2.bit.INT1SEL = 12;  // ADCCH12 (ADC-B4) EOC
uzrokuje ADCInterrupt 1
      AdcRegs.INTSEL1N2.bit.INT1CONT = 1;  // Postavi ADCInterrupt
1 na auto brisanje
      AdcRegs.INTSEL1N2.bit.INT1E = 1;     //     Omogući     ADC
interrupt 1
      AdcRegs.ADCINTSOCSEL1.bit.SOC0  =  1;    //  ADCInterrupt  1
Uzrokuje start SOC0 a i svih ostalih ispod. Zbog toga je broj
jedan svugdje.
      AdcRegs.ADCINTSOCSEL1.bit.SOC1 = 1;
```

```
      AdcRegs.ADCINTSOCSEL1.bit.SOC2 = 1;
      AdcRegs.ADCINTSOCSEL1.bit.SOC4 = 1;
      AdcRegs.ADCINTSOCSEL1.bit.SOC5 = 1;
      AdcRegs.ADCINTSOCSEL1.bit.SOC6 = 1;
      AdcRegs.ADCINTSOCSEL2.bit.SOC8 = 1;
      AdcRegs.ADCINTSOCSEL2.bit.SOC9 = 1;
      AdcRegs.ADCINTSOCSEL2.bit.SOC10 = 1;
      AdcRegs.ADCINTSOCSEL2.bit.SOC12 = 1;
      AdcRegs.ADCINTSOCSEL2.bit.SOC14 = 1;

// Postavi kanal za pretvorbu kada je SOCx primljen
      AdcRegs.ADCSOC0CTL.bit.CHSEL= 0;//  Pretvori  ADC-A0  (CH0)
kada je SOC0 primljen
      AdcRegs.ADCSOC1CTL.bit.CHSEL= 1;//  Pretvori  ADC-A1  (CH1)
kada je SOC1 primljen
      AdcRegs.ADCSOC2CTL.bit.CHSEL= 2;
      AdcRegs.ADCSOC4CTL.bit.CHSEL= 4;
      AdcRegs.ADCSOC5CTL.bit.CHSEL= 5;
      AdcRegs.ADCSOC6CTL.bit.CHSEL= 6;
      AdcRegs.ADCSOC8CTL.bit.CHSEL= 8;
      AdcRegs.ADCSOC9CTL.bit.CHSEL= 9;//  Pretvori  ADC-B1  (CH9)
kada je SOC9 primljen
      AdcRegs.ADCSOC10CTL.bit.CHSEL= 10;
      AdcRegs.ADCSOC12CTL.bit.CHSEL= 12;
      AdcRegs.ADCSOC14CTL.bit.CHSEL= 14;

      EDIS;

      AdcRegs.ADCSOCFRC1.all = 0x1000;      // Omogući start ADC-
a uzrokovanog SOC12 događajem

//    Beskonačna petlja

      for(;;)  //
      {
                  AdcResults[0]    =    AdcResult.ADCRESULT0;    //
Rezultati pretvorbe prvog od 11 ADC kanala
                  AdcResults[1] = AdcResult.ADCRESULT1;
                  AdcResults[2] = AdcResult.ADCRESULT2;
                  AdcResults[3] = 0;
      // ADC-A3 ne postoji na controlSTICK
                  AdcResults[4] = AdcResult.ADCRESULT4;
                  AdcResults[5] = AdcResult.ADCRESULT5;
                  AdcResults[6] = AdcResult.ADCRESULT6;
                  AdcResults[7] = 0;
      // ADC-A7 ne postoji na controlSTICK
                  AdcResults[8] = AdcResult.ADCRESULT8;
                  AdcResults[9] = AdcResult.ADCRESULT9;
                  AdcResults[10] = AdcResult.ADCRESULT10;
                  AdcResults[11] = 0;
      // ADC-B3 ne postoji  controlSTICK
                  AdcResults[12] = AdcResult.ADCRESULT12;
                  AdcResults[13] = 0;
      // ADC-B5 ne postoji na controlSTICK
                  AdcResults[14] = AdcResult.ADCRESULT14;
```

```
                AdcResults[15] = 0;
      // ADC-B7 ne postoji na controlSTICK
      }

} //Kraj glavnog koda
```

8.9. Programski kod za PWM okidnu ADC pretvorbu [6]

```
#include "PeripheralHeaderIncludes.h"
#include "F2806x_EPwm_defines.h"

// Funkcijski prototipovi

void DeviceInit(void);
void InitFlash();
void MemCopy(Uint16 *SourceAddr, Uint16* SourceEndAddr, Uint16*
DestAddr);

// Općenita deklaracija varijabli

extern      Uint16      RamfuncsLoadStart,      RamfuncsLoadEnd,
RamfuncsRunStart;
Uint16 adc_result;
Uint16 duty_cycle;


// Početak glavnog koda

void main(void)
{
      Uint16 i = 0;

//    Generalna inicijalizacija

      DeviceInit();
#ifdef FLASH
      MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd,
&RamfuncsRunStart);

      InitFlash();
#endif //(FLASH)

// PWM inicijalizacija za upotrebu pina 31(PWM1A-DAC) kao ADC
ulaznog signala

#define period 40      // 2MHz kada je PLL postavljen na 0x10
(80MHz)

      duty_cycle = 20;              // Inicijaliziraj duty na 50%
```

```
  // Vremensko bazni registri
      EPwm1Regs.TBPRD = period;           //     Postavi     period
tajmera, PWM frequency = 1 / period
      EPwm1Regs.TBPHS.all = 0;            //    Vremensko    bazni
registar
      EPwm1Regs.TBCTR = 0;       //   Vremensko   bazni   brojački
registar
     EPwm1Regs.TBCTL.bit.PRDLD  =  TB_IMMEDIATE;        //Postavi
automatsko dizanje
     EPwm1Regs.TBCTL.bit.CTRMODE   =   TB_COUNT_UP;   //   Uzlazni
brojački mod za asimetrični PWM
      EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;       //Onemogući
dizanje faze
      EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_DISABLE;
      EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
      EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;

      EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
      EPwm1Regs.CMPCTL.bit.LOADAMODE     =     CC_CTR_ZERO;     //
Postavljanje usporedbenih vrijednosti
      EPwm1Regs.CMPA.half.CMPA = duty_cycle;       //     Početno
postavi duty na 50%


// Postavljanje akcija

      EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;  // Postavi PWM1A na nulti
događaj
      EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;        // Obriši PWM1A
na događaj A,uzlazno brojanje

// ADC Inicijalizacija

    EALLOW;
      AdcRegs.ADCCTL1.bit.ADCREFSEL   = 0;
      AdcRegs.ADCCTL1.bit.ADCBGPWD    = 1;
      AdcRegs.ADCCTL1.bit.ADCREFPWD   = 1;
      AdcRegs.ADCCTL1.bit.ADCPWDN     = 1;
      AdcRegs.ADCCTL1.bit.ADCENABLE   = 1;
      for(i=0; i<5000; i++){}

      AdcRegs.ADCCTL1.bit.INTPULSEPOS = 1;

      AdcRegs.INTSEL1N2.bit.INT1SEL = 0;  // ADCCH0 (ADC-A0) EOC
uzrokuje ADCInterrupt1
      AdcRegs.INTSEL1N2.bit.INT1CONT = 1;        //       Postavi
ADCInterrupt 1 na auto brisanje(neprekidna pretvorba)
      AdcRegs.INTSEL1N2.bit.INT1E = 1;      //             Omogući
ADCInterrupt1; 0=ništa se ne događa, 1=ADCInt1, 2=ADCInt2

      AdcRegs.ADCINTSOCSEL1.bit.SOC0 = 1;        // ADCInterrupt1
uzrokuje SOC0
      AdcRegs.ADCSOC0CTL.bit.CHSEL= 0;           // Pretvori ADC-
A0 (CH0) kada SOC0 bude primljen
      AdcRegs.ADCSOC0CTL.bit.ACQPS = 8;          //  Postavi  S/H
prozor na 8 ciklusa sata (112.5ns)
```

```
      AdcRegs.ADCSOCFRC1.all = 0x0001;           //  Startaj  ADC
uzrokovan ADCInterrupt1 događajem
      EDIS;

//    Beskonačna petlja

      for(;;)
      {
                  // adc_result su između 0x0000 and 0x0FFF ( Od 0
do 3.3 V)
                  adc_result = AdcResult.ADCRESULT0;         //
Pohrani ADC mjerenja u adc_result
                  EPwm1Regs.CMPA.half.CMPA   =   duty_cycle;     //
Ažuriraj duy_cycle

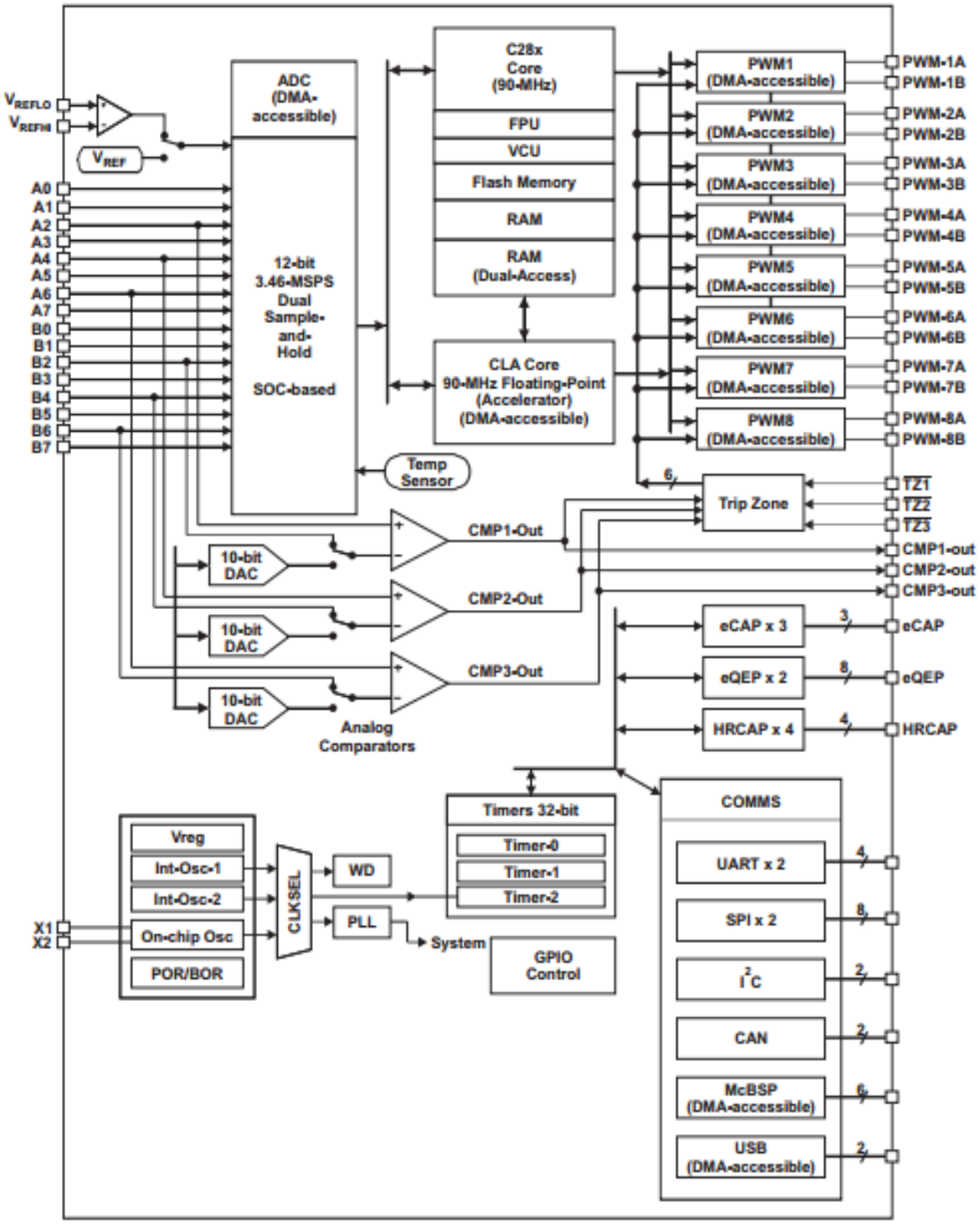
      }

      // Dodaj  "duty_cycle" i "AdcResults" u watch window da bi
se mijenjao DAC izlaz te promatrali rezultati pretvorbe

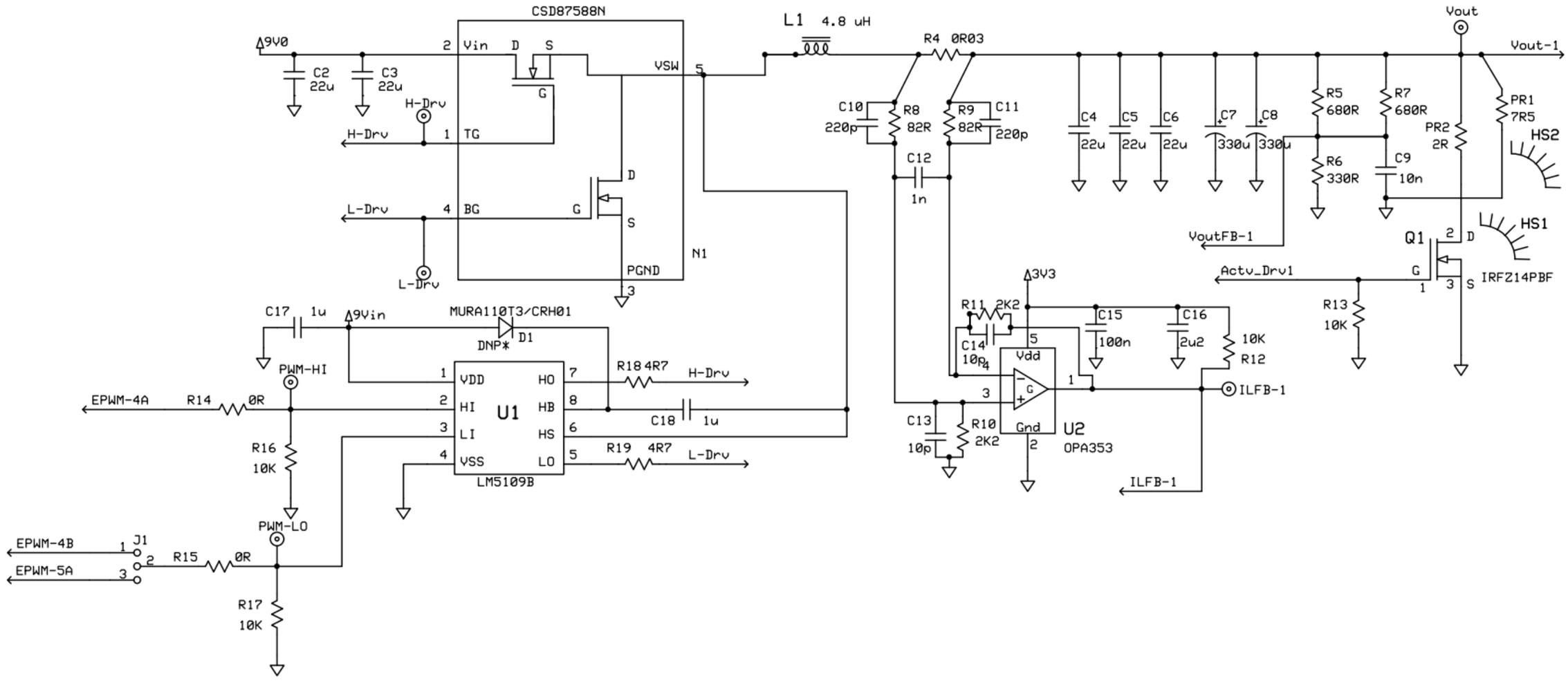


}
```

8.10. Dijagram sustava porodice TMS320F2806x mikrokontrolera [2]



8.11. Električna shema silaznog digitalnog pretvarača (Digital Power BoosterPack) [7]



9. POPIS SLIKA

Slika 2.1 Sklopovski razvojni sustav F28069 controlSTICK 2
Slika 2.2 Pojednostavljeni blok dijagram TMS320F28069 mikrokontrolera [5] 4
Slika 2.3 Funkcijski blok dijagram TMS320F28069 mikrokontrolera [2] 4
Slika 2.4 Prikaz memorijske mape [5] 8
Slika 3.1 Odabir radnog područja zvanog Workspace 13
Slika 3.2 Otvaranje postojećih projekata 13
Slika 3.3 Povezivanje CCS-a sa mikrokontrolerom 14
Slika 3.4 Aktiviranje projekta 14
Slika 3.5 Prikaz prozora za promatranje i mijenjanje varijabli 15
Slika 3.6 Uključenje moda za rad u realnom vremenu 15
Slika 3.7 Prikaz tipki za upravljanje i uklanjanje projekta 15
Slika 4.1 Programski kod akcijskog kvalifikacijskog modula za dva simetrična PWM-a [6] 17
Slika 4.2 Princip stvaranja simetričnih PWM signala [6] 18
Slika 4.3 Programski kod za dodavanje varijabli u watch window [6] 18
Slika 4.4 Varijable prikazane u watch window 19
Slika 4.5 Simetrični PWM signali frekvencije 80 kHz 19
Slika 4.6 Simetrični PWM signali frekvencije 40 kHz 20
Slika 4.7 Programski kod akcijskog kvalifikacijskog modula za dva asimetrična PWM-a [6] 22
Slika 4.8 Princip stvaranja asimetričnih PWM signala [6] 22
Slika 4.9 Asimetrični PWM signali frekvencije 80 kHz 23
Slika 4.10 Asimetrični PWM signali neovisno modulirani 24
Slika 4.11 Konfiguracijske opcije mrtvo-zonskog podmodula [6] 26
Slika 4.12 Prikaz varijabli za mijenjanje rastućih i padajućih bridova PWM-a 27
Slika 4.13 Prikaz reguliranja mrtve zone kod PWM signala prikazanog žutom bojom . 27
Slika 4.14 Prikaz varijabli za namiještanje trip zone 30
Slika 4.15 Dva asimetrična PWM signala bez efekta trip zone 30
Slika 4.16 Efekt trip zone 31
Slika 4.17 Programski kod akcijskog kvalifikacijskog modula 32
Slika 4.18 Prikaz stvaranje PWM signala u uzlaznom brojačkom modu [6] 32

Slika 4.19 Operacijska logika koju koristi MEP tehnologija [1] 33
Slika 4.20 Programski kod za omogućavanje visokorezolucijskih registara i kontrolu rastućeg brida [6] 33
Slika 4.21 Programski kod za ažuriranje usporedbenog A registra [6] 34
Slika 4.22 Brojački usporedbeni visokorezolucijski A registar [6] 34
Slika 4.23 Komparatorski blok dijagram [1] 36
Slika 4.24 Komparator [1] 36
Slika 4.25 Programski kod za mijenjanje unutarnjeg napona te napona na EPWM1A izlazu [6] 37
Slika 4.26 Programski kod za mijenjanje unutarnje DAC vrijednosti napona [6] 39
Slika 4.27 Prikaz generiranja dva simetrična PWM signala 39
Slika 4.28 Efekt trip zone temeljen na usporedbi dva analogna napona 40
Slika 4.29 Blok dijagram ADC pretvarača [1] 42
Slika 4.30 Neprekidna ADC pretvorbena sekvenca [6] 43
Slika 4.31 Stvaranje ADCinterupta1 ciklus prije kraja pretvorbe [6] 43
Slika 4.32 Programski kod za startanje kontinuirane ADC pretvorbe [6] 44
Slika 4.33 Stvaranje PWM signala i mjesto početka pretvorbe [6] 46
Slika 4.34 Konfiguriranje ADC okidača u Round robin modu [6] 47
Slika 4.35 Prikaz pretvorene digitalne vrijednosti za određeno vrijeme vođenja PWM signala 47
Slika 5.1 Digitalni silazni pretvarač 48
Slika 5.2 Glavne komponente silaznog digitalnog pretvarača [7] 49
Slika 5.3 Pojednostavljena struktura silaznog pretvarača [7] 50
Slika 5.4 Naponski mod upravljanja pretvaračem 51
Slika 5.5 Nadstrujna zaštita pretvarača u naponskom modu 52

10. POPIS TABLICA

Tablica 4.1 Prikaz sklopovskog povezivanja mikrontrolera..... 16
Tablica 4.2 Prikaz sklopovskog povezivanja mikrokontrolera..... 21
Tablica 4.3 Prikaz sklopovskog povezivanja mikrokontrolera..... 24
Tablica 4.4 Postavka sklopovlja mikrokontrolera za efekt trip zone..... 28
Tablica 4.5 Prikaz sklopovskog povezivanja mikrokontrolera..... 31
Tablica 4.6 Prikaz sklopovskog povezivanja mikrokontrolera..... 35
Tablica 4.7 Tablica istine komparatora..... 36
Tablica 4.8 Prikaz sklopovskog povezivanja mikrokontrolera..... 38
Tablica 4.9 Prikaz sklopovskog povezivanja mikrokontrolera..... 40
Tablica 4.10 Prikaz sklopovskog povezivanja mikrokontrolera..... 45