Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007.52
DOI: https://doi.org/10.24867/25IH01Milojicic

IMPLEMENTACIJA BOOTLOADER-A ZA MIKROKONTROLER STM32F413ZH

BOOTLOADER IMPLEMENTATION FOR STM32F413ZH MICROCONTROLLER

Žarko Milojičić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – MEHATRONIKA, ROBOTIKA I AUTOMATIZACIJA

Kratak sadržaj – *U ovom radu opisana je arhitektura i način implementacije bootlaoder-a za STM32 mikrokontroler, kao i način njegovog testiranja.*

Ključne reči: *Bootloader, embedded sistemi, mikrokontroleri*

Abstract – *This paper describes the architecture and implementation method of the bootloader for the STM32 microcontroller as well as the method of its testing.*

Keywords: *Bootloader, embedded systems, microcontrollers*

1. UVOD

U toku razvoja embedded sistema često se dešava da pored raznih testiranja tokom razvoja, softver sadrži greške koje dovode do nepouzdanog rada sistema. Sam proces vraćanja uređaja proizvođaču kako bi se greška ispravila bi u određenim slučajevima bio jako neisplativ po pitanju vremena i novca. Iz tog razloga je projektovan bootloader-a koji omogućava ažuriranje trenutnog softvera bez potrebe za kontaktiranjem proizvođača.

Bootloader je aplikacija koja omogućava da se softver embedded sistem ažurira bez korišćenja dodatnog specijalizovanog hardvera kao što je npr. JTAG programator.

Da bi se softverski sistem mogao ažurirati, bootloader mora na neki način da primi novu verziju softvera (binarni fajl tj. image aplikacije), a to se može postići korišćenjem različitih protokola kao što su: UART, I2C, Ethernet, USB, WIFI, Bluetooth, itd. [1].

Bootloader se ne razlikuje mnogo od standardnih aplikacija. Jedina razlika je u tome što deli memorijski prostor sa drugom aplikacijom i to što ima sposobnost da drugu aplikaciju izbriše iz memorije ili da na njeno memorijsko mesto upiše neku drugu aplikaciju.

Osim što mora da ima mehanizme koji omogućavaju primanje novog image-a (binarni fajl programa) i njegovo smeštanje u memoriju, bootloader takođe mora da ima sposobnost validacije novog image-a. Ta validacija podrazumeva proveru integriteta i autentičnosti novog primljenog image-a ili image-a koji treba pokrenuti.

2. ZAHTEVI BOOTLOADER-a

Pored osnovnih funkcija koje bootloader treba da ispuni postoje i dodatni zahtevi koji se mogu svesti na sledeće:

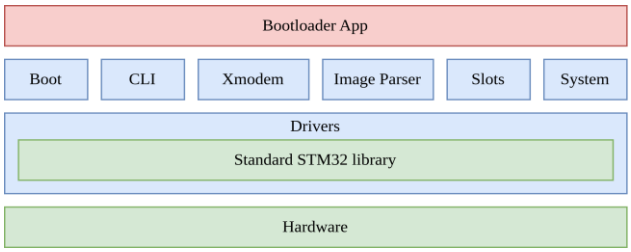
1. Čuvanje više image-a aplikacija.
2. Primanje novog image-a putem UARTa.
3. Provere integriteta image-a.
4. Mora postojati aplikacija koja će se pokretati kada se uređaj uključuje prvi put nakon proizvodnje ili kad integritet svih ostalih image-a nije validan.
5. Omogućiti korisnički interfejs korisniku u slučaju neuspešnog pokretanja imaga-a, koji će ponuditi opciju: kopiranja, brisanja, ažuriranja.
6. Prikaz informacija o image-ima koji se nalaze u memoriji.
7. Omogućiti ažuriranje aplikacije prilikom novog pokretanja, ukoliko postoji novi image.
8. Memorijski prostor u kome se nalazi bootloader mora biti zaštićen prilikom rada korisničke aplikacije.

Jedan od zahteva je da se bootloader mora implementirati za mikrokontroler STM32F413ZHT koji ima sledeće karakteristike:

- Procesor Arm Cortex-M4.
- 1,5 MB of Flash.
- 320 KB of SRAM.
- Clock 100MHz.

2. ARHITEKTURA BOOTLOADER-A

Na osnovu postavljenih zahteva u prethodnom poglavlju, došlo se do arhitekture bootlaoder-a koja je prikazana na sledećoj slici (slika 1).



Slika 1. *Arhitektura bootloader-a.*

Kao što se može videti na slici, sam bootloader koristi sledeće module:

- Slot system – odgovoran za sve operacije koje se vrše nad fleš memorijom, kao što je upis novih podataka, kopiranje, brisanje i čitanje podataka iz fleš memorije. Ovaj modul biće korišćen i od strane korisničke aplikacije. Modul deli fleš memoriju na slotove u kojima će se smeštati image-i aplikacija:

1. Bootloader (128 kB) – Čuva bootloader aplikaciju.
2. Izvršni slot (exe slot) (256 kB) – Čuva image koja treba da se izvrši.
3. Slot za ažuriranje (update slot) (256kB) – Čuva image koja se dobija tokom ažuriranja.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bila dr Gordana Ostojić, red. prof.

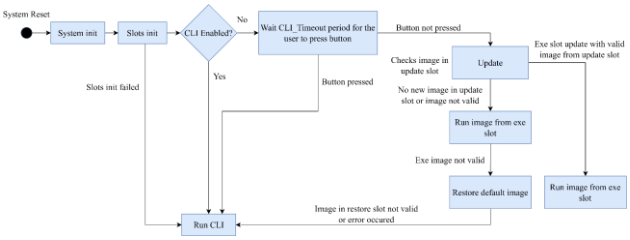
4. Fabrički slot (default slot) (256kB) – Čuva fabrički image (recover image).
5. Prazan slot 1 (empty slot 1) (256kB) – Čuva backup image.
6. Prazan slot 2 (empty slot 2) (256kB) – Prazan slot (za slučaj nekih drugih potreba korisnika).

- Command line interface – Modul koji omogućava komunikaciju između uređaja i računara putem UART-a, omogućava korisniku da pošalje komande uređaju koje će potom uređaj izvršiti.
- Xmodem – Modul koji omogućava slanje image-a ka uređaju koristeći UART Xmodem protokol.
- Image parser – Izdvaja informacije iz image-a o samom image-u.
- Boot – Modul koji upravlja svim operacijama nad image-ima koristeći slot modul i image parser, vrši proveru integriteta aplikacije koje treba pokrenuti.
- System – Modul koji inicijalizuje sve periferije neophodne za rad i jedinicu za zaštitu memorije.
- Driver – deo standardne STM32 biblioteke koje se koriste od strane ostalih modula.

Ovakvom arhitekturom pokriveni su svi zahtevi bootloader-a koji su bili postavljeni.

2. OPŠTI OPIS FUNKCIONALNOSTI BOOTLOADER-A

Algoritam rada bootloader-a do kojeg se došlo na osnovu postavljenih zahteva je prikazan na slici 2. Osim što ispunjava sve zahteve, algoritam omogućava i lakše testiranje prilikom razvoja dodavanjem CLI_enabled flag-a. Ovaj flag je uključen samo prilikom razvoja kako bi se funkcionalnosti lakše testirale, dok se u produkciji on isključuje.



Slika 2. *Algoritam rada bootloader-a.*

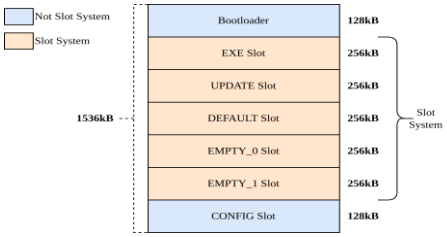
Prilikom uključenja uređaja i nakon inicijalizacije sistema, korisnički meni biće prikazan korisniku ako je CLI_Enabled flag uključen.
Ako je komandni korisnički interfejs (CLI) onemogućen, korisniku se daje nekoliko sekundi da pritisne bilo koje slovo na tastaturi kako bi se CLI pokrenuo. Ako korisnik ne pokrene CLI, bootloader će prvo proveriti da li postoji novi image u slotu za ažuriranje. Ako je image validan (CRC je tačan), napraviće se rezervna kopija image-a izvršnog slota (ako je omogućena backup opcija) kopiranjem image-a iz izvršnog slota u backup slot. Ako nema image-a u izvršnom slotu, pravljenje rezervne kopije će biti preskočeno.
Sledeći korak jeste kopiranje slota za ažuriranje u izvršni slot. Pre kopiranja image-a iz izvornog slota u odredišni slot, odredišni slot treba da se obriše i to se radi svaki put. Kada se novi image uspešno premesti u izvršni slot, slot za ažuriranje se briše. Pre pokretanja novog image-a iz izvršnog slota, proverava se njegov magični broj i CRC, ako je ispravan novi image će se pokrenuti, ako nije, image iz fabričkog slota će biti kopiran u izvršni slot i biće pokrenut.
Pre samog pokretanja image-a iz izvršnog slota vrši se zaštita slota u kojem se nalazi bootloader aplikacija, fabričkog slota kao i zaštita svih header-a slotova u kojima se nalaze informacije o njima. Ukoliko korisnička aplikacija pokuša upis u te slotove tokom rada, hard fault će se desiti.

3. MODULI BOOTLOADER-A

U ovom poglavlju biće opisani moduli koji su korišćeni u bootloader-u kako bi se bolje razumelo njegovo funkcionisanje.

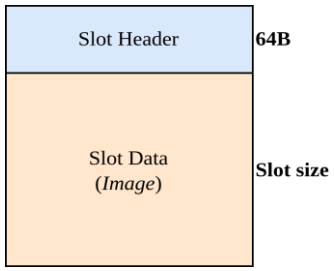
3.1 Slot sistem

Slot sistem predstavlja modul koji upravlja fleš memorijom uređaja, tj. upravlja slotovima u kojima su smeštene image-i aplikacija. Slot sistem obavlja funkciju upisa, čitanja i brisanja fleš memorije pri tome koristeći STM32 HAL biblioteku.
Svakom slotu dodeljeno je 256kB memorijskog prostora. Izgled slot sistema može se videti na slici 3.



Slika 3. *Slot sistem u fleš memoriji.*

Kao logička celina, svaki slot se sastoji iz dva dela, slot header-a i podataka slota, kao što je prikazano na slici 4. Sam slot header sadrži sve potrebne informacije o slotu, a podaci slota predstavljaju sam image aplikacije.



Slika 4. *Slot sistem u fleš memoriji.*

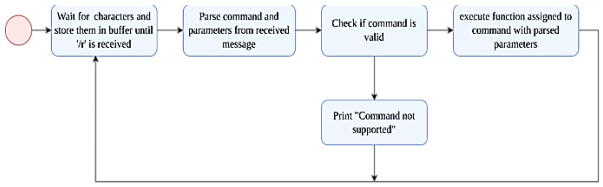
Slot header je veličine 64B, i on sadrži sledeće informacije:

1. Magični broj slota (4B) – Koristi se za proveru da li slot uopšte postoji.
2. Ime slota (32B) – Naziv slota.
3. Tip slota (4B) – Govori da li je u slotu image aplikacije ili podatak koji nije image.
4. Veličina slota (4B) – Predstavlja veličinu slota.
5. Dozvola slota (2B) – Govori koje operacije mogu biti izvršene nad slotom, upis/čitanje ili samo čitanje slota.
6. Status korišćenja (2B) – Govori da li je slot zauzet ili nije.

3.2 Komandni korisnički interfejs

Da bi korisnik mogao da šalje komande uređaju sa računara i da bi ih uređaj izvršavao, potrebno je uvesti komandni korisnički interfejs (Command line interface -

CLI). U ovom slučaju, UART će se koristiti za komunikaciju između računara i uređaja.
Korisnik šalje komandu sa parametrima, ako komanda postoji ona se izvršava, ako ne postoji korisnik će o tome biti obavešten. Algoritam komandnog korisničkog interfejsa prikazan je na slici 5.



Slika 5. *Slot sistem u fleš memoriji.*

3.3 XMODEM Protokol

XMODEM je UART protokol za prenos datoteka koji omogućava korisnicima da prenose datoteke između različitih uređaja. XMODEM deli podatke u niz paketa koji se šalju prijemnoj strani. Takođe sadrži dodatne informacije koje omogućavaju primaocu da utvrdi da li je taj paket ispravno primljen ili ne. Paket se ponovo šalje nakon što ga traži primalac ako naiđe na grešku. Transfer se prekida u slučaju primljenog određenog broja paketa koji nisu validni [2]. Sadržaj paketa prikazan je u tabeli 1.

Tabela 1. *Sadržaj paketa Xmodem protokola.*

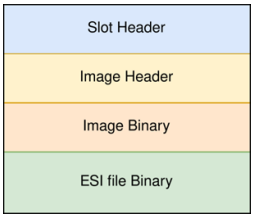
Header	Packet number	1's Compliment of packet number	The packet	CRC16
1B	1B	1B	128B	2B

3.3 Image parser

Image parser je modul koji se koristi za raščlanjivanje image header-a. Informacije iz header-a se koriste za uspešno izvršenje procesa pokretanja.
Image header sadrži sledeće informacije:

- Magični broj – služi za proveravanje da li slika uopšte postoji.
- Veličina slike
- Ime slike – maksimalan broj karaktera u imenu je 32.
- Verzija slike – Formata X.X
- Veličina ESI fajla – veličina binarnog fajla koji je smešten iza slike.
- Prazni bajtovi – u slučaju dodatnih potreba i modifikacija.
- Potpis – U ovom prostoru biće smeštena CRC32 vrednost koja je izračunata na osnovu binarnih vrednosti same slike i ESI fajla od strane korisnika.
- Dopunski bajtovi - bajtovi koji se dodaju da bi početak vektor tabele (sam početak image-a) bio poravnat na 0x200 (Adresa početka vektor tabele mora biti deljiva sa 0x200). Ovo predstavlja uslov za pravilno pokretanje programa postavljenog od strane STM-a.

Nakon uvođenja image header-a sam slot će izgledati ovako:



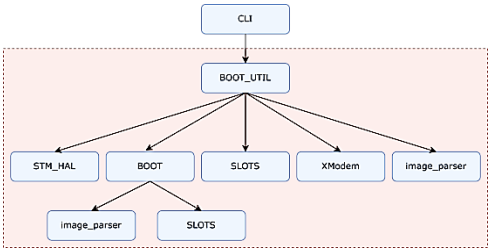
Slika 6. *Prikaz celog slota.*

3.4 Boot modul

Boot modul upravlja svim operacijama nad image-om koristeći funkcije slot modula i image parsera . Modul za pokretanje se sastoji od dve celine:

1. boot – Upravlja svim operacijama nad image-ima u slotovima.
2. boot_util – Predstavlja wrapper oko boot modula i korišćen je od strane CLI modula.

Veza između boot i boot_util modula je prikazana na slici 7. Vrh strelice označava modul koji se poziva od strane modula koji se nalazi na početku strelice:



Slika 7. *Veza između modula.*

Boot modul može da izvodi sledeće operacije:

- Pokretanje image-a iz izvršnog slota.
- Provera integriteta image-a u slotu koristeći CRC32 algoritam.
- Pokretanje image-a iz određenog slota – image se prvo kopira iz slota u izvršni slot, odakle se pokreće.
- Ažuriranje izvršnog slota – Image iz slota za ažuriranje se kopira u izvršni slot, nakon čega se image u slotu za ažuriranje briše, zatim se image u izvršnom slotu pokreće ako je validan.
- Pokretanje image-a iz fabričkog slota – Image se kopira iz fabričkog slota u izvršni slot, nakon čega se image u izvršnom slotu pokreće.

Boot_util modul implementira sledeće komande koje se prikazuju preko CLI-a korisniku:

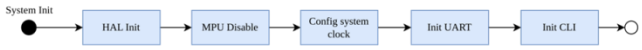
1. List – Ova opcija prikazuje koji slot je iskorišćen, veličinu image-a u njima i verziju aplikacije u svakom slotu.
2. Boot – Pokreće aplikaciju iz izabranog slota (image iz željenog slota biće kopiran u izvršni slot odakle će biti pokrenut).
3. Format – Briše image iz izabranog slota.
4. Update – kopira image iz update slota u exe i pokreće ga.
5. Restore – Image iz fabričkog slota se kopira u izvršni slot, odakle se pokreće.
6. Copy – Kopira image iz jednog slota u drugi.
7. sendImage – Ažuriranje putem UART-a se pokreće, uređaj očekuje početak slanja image-a od strane računara.
8. FormatSlots – Kreira ceo slot sistem. Koristi se u slučaju kada je slot sistem oštećen.
9. Reset– resetuje uređaj.

Prikaz svih CLI komandi koje su omogućene korisniku mogu se videti na slici 8:

Slika 8. *CLI komande*

3.4 System modul

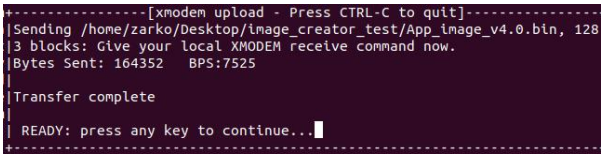
System modul se koristi za inicijalizaciju svih potrebnih periferija i jedinice za zaštitu memorije. Njegova init funkcija se prva poziva tokom pokretanja bootloader-a. Procedura pokretanja je prikazana na slici 9:



Slika 9. *Procedura inicijalizacije*

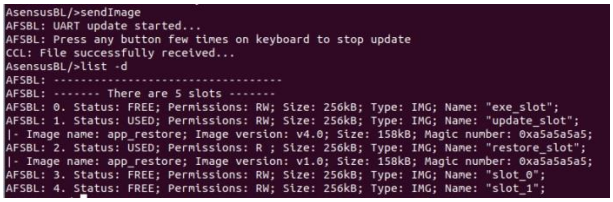
4.0 PROCES AŽURIRANJA APLIKACIJE

Proces ažuriranje se pokreće pozivom komane sendImage, nakon koje bi iz terminala trebalo da se inicira početak slanja. Status terminala koji je uspešno poslao image-a prikazan je na slici 10:



Slika 10. *Poruka terminala nakon uspešno poslatog image-a.*

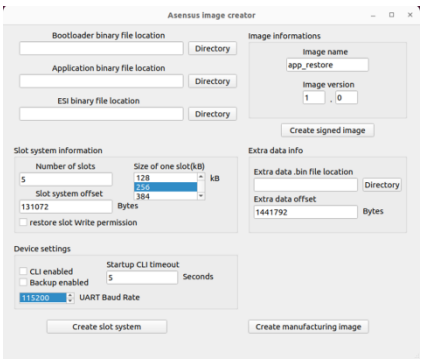
Nakon uspešnog prijema image-a, novi image se može videti u slotu za ažuriranje.



Slika 11. *Prikaz slotova nakon primljenog image-a.*

5. APLIKACIJA ZA FORMIRANJE FABRIČKOG IMAGE-A

Na slici 12 prikazan je interfejs aplikacije koja se koristi za kreiranje image-a kojim će se flešovati uređaj prilikom proizvodnje. Osim celog image-a koji će sadržati bootloader, slot sistem sa image-om u restore slotu i dodatne podatke, aplikacija ima mogućnost da kreira posebno potpisani image (koji se koristiti prilikom ažuriranja) i image koji sadrži samo slot sistem bez bootloader-a i ostalih celina.



Slika 12. *Interfejs aplikacije za kreiranje image-a.*

6. TESTIRANJE

Testiranje je izvršeno tako što je svaka komanda testirana sa različitim parametrima. Pored ovog testiranja izvršeno je testiranje slučajeva koji nisu tipični kao što je prekidanje napajanja tokom ažuriranja i kopiranja image-a, kao i prikadanje komnikacije tokom slanja image-a. U ovim slučajevima bootloader ne sme pokrenuti image koji je oštećen, što znači da bi bilo koja aplikacija, koja se pokrene, morala da radi normalno.

7. ZAKLJUČAK

Na osnovu postavljenih zahteva došlo se do arhitekure bootloader-a kojom su identifikovani potrebni moduli, a nakon toga i algoritam rada.

Iz razloga što je bootloader jedan od bitnijih komponenti embedded sistema potrebno ga je temeljno istestirati na tipične i ne tipične slučajeve, pritom se dodatno finalno testiranje prepušta trećem licu koje može proći kroz još veći broj slučajeva.

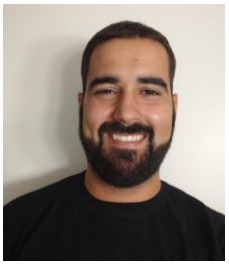
Ovim su obuhvaćena sva poglavlja koja opisuju razvoj i testiranje bootloader-a jednog embedded sistema.

8. LITERATURA

[1] Jacob Beningo, Bootloader Design for Microcontrollers in Embedded Systems, 2015.

[2] Internet stranica https://www.techopedia.com/definition/1886/xmodem (pristupljeno u avgustu 2023)

Kratka biografija:



Žarko Milojičić rođen je 10.02.1996. god. u Kraljevu. Osnovnu i srednju školu završio je u Kraljevu. Osnovne akademske studije završio je na Fakultetu tehničkih nauka 2019. god. i nakon toga upisao master akademske studije na istom fakultetu. Master rad iz oblasti Mehatronika, robotika i automatizacija odbranio je 2023. god. Kontakt: zarko.kv036@gmail.com