Elektrotehnički fakultet, Univerzitet u Beogradu

Predmet : Integrisani računarski sistemi

- Projektni zadatak -

Implementacija parking senzora pomoću mikrokontrolera

Profesor:

Nenad Jovičić

Nikola Cvetković

Student:

Aleksandar Milošević 658/17

Beograd 2022.

Sadržaj

Uvod

- Projektni zadatak

Upotrebom razvojne platforme MSP430F5529LP bilo je potrebno realizovati komunikaciju sa ultrazvučnim senzorom HC-SR04 koji meri udaljenost prepreke ispred sebe. Izmereni rezultat se prikazuje na sedmo-segmentnom displeju. Diode LED3 i LED4 se koriste da bi signalizirale koliko je predmet udaljen. Pritiskom tastera, izmereni rezultat se šalje putem putem USB interfejsa na računar.

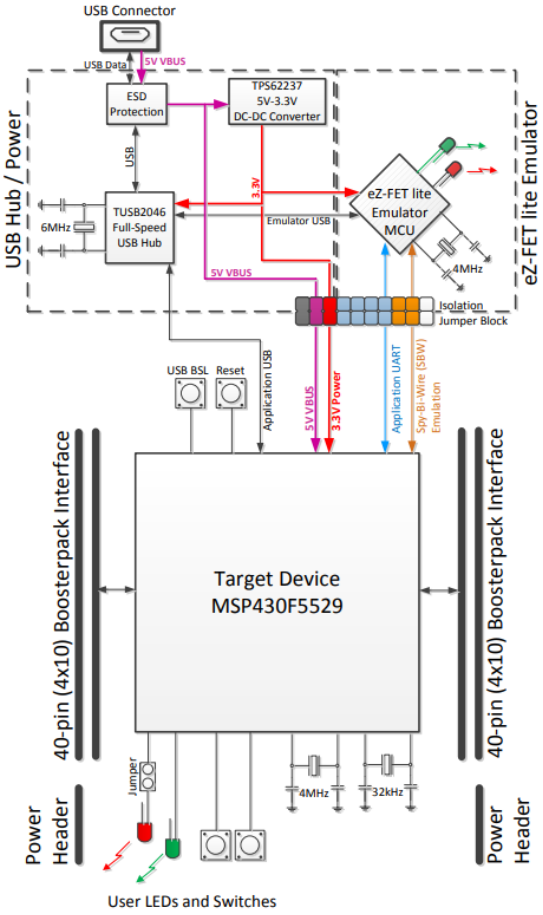
- Karakteristike razvojne platforme MSP430F5529LP

Razvojna platforma MSP430F5529LP sadrži MSP430F5529 mikrokontroler iz familije MSP430 mikrokontrolera (MCU - Microcontroller Unit) koje proizvodi kompanija Texas Instruments. On poseduje 16-bitni procesor (CPU - Central Processing Unit) sa redukovanim setom instrukcija (RISC - Reduced Instruction Set Computer), sa Von-Neumann-ovom arhitekturom sa dve magistrale MAB i MDB koje dele isti adresni prostor. Dakle on u sebi poseduje 27 osnovnih instrukcija i pruža mogućnost 7 vrsta adresiranja. Sam CPU sadrži 16 internih registara, od kojih su praktično 12 njih opšte namene (R4 do R15), dok se dva koriste kao PC (Program Counter), SP (Stack Pointer), i još dva kao generatori konstanti, ali je takođe dozvoljeno čitanje i upis u njih. Takođe se ovakva MCU ubraja u Low Power MCU-ove koji imaju smanjenu potrošnju i omogućava pet Low Power režima. Mogućnost ulaska u neki od LPM se ogleda u ukidanju signala takta za određene delove sistema, čime se smanjuje sama potrošnja celokupnog sistema. Takođe inicijalizacija svih I/O pinova da rade u output režimu dodatno može smanjiti potrošnju (jer se isključuju schimit trigger kola), jer su po default-nom režimu svi pinovi u input režimu. Memorijska mapa mikrokontrolera sadrži Flash/ROM (u koju se može smeštati i programski kod i podaci), RAM (u koju se smeštaju privremeni podaci) i adrese namenjene za pristup internim periferijama mikrokontrolera. Ukoliko bi programski kod bio smešten u RAM memoriji, sistem bi trošio skoro duplo manje energije, nego kada je programski kod smešten u Flash memoriji. Sam procesor se taktuje učestanošću 1MHz.

Slika 1 - MSP430F5529 mikrokontroler

Sam proces razvoja ovakvog sistema podrazumeva cross-development razvoj softvera. Cross-development omogućava da se softver razvija (piše, build-uje, kompajlira) na host sistemu, a potom putem USB ili nekog drugog interfejsa se ceo programski kod prenosi (smešta u odgovarajuću memoriju) na target sistemu. Host sistem je sistem na kome se razvija kod (u ovom slučaju PC računar), a target sistem je sistem na kome se pokreće taj kod (u ovom slučaju MSP430F5529 mikrokontroler).

Za realizaciju softvera sistema, korišćeno je Code Composer Studio 11.1 (CCS11.1) integrisano razvojno okruženje (IDE - Integrated Development Environment), kompanije Texas Instruments, bazirano na *eclipse* platformi. Ono omogućava cross-development razvoj (razvoj programskog koda na PC računaru i prenos putem odgovarajućeg interfejsa na sistem sa MCU-om). Programski kod je pisan na C programskom jeziku.



Slika 2 - blok šema MSP430F5529LP platforme

- Upotrebljene periferije

Na razvojnu platformu MSP430F5529LP povezane su i druge eksterne periferije putem dodatnog (plavog) shield-a. Takođe na samu MSP430F5529LP se povezuje i HC-SR04 ultrazvučni senzor preko isturenih pinova.

Eksterne:

- HC-SR04 ultrazvučni senzor
- dva 7-segmentna displeja
- 4 LED diode (LED1, LED2, LED3, LED4)
- 3 tastera (S1, S2, S3)
- USB konektor

Interne:

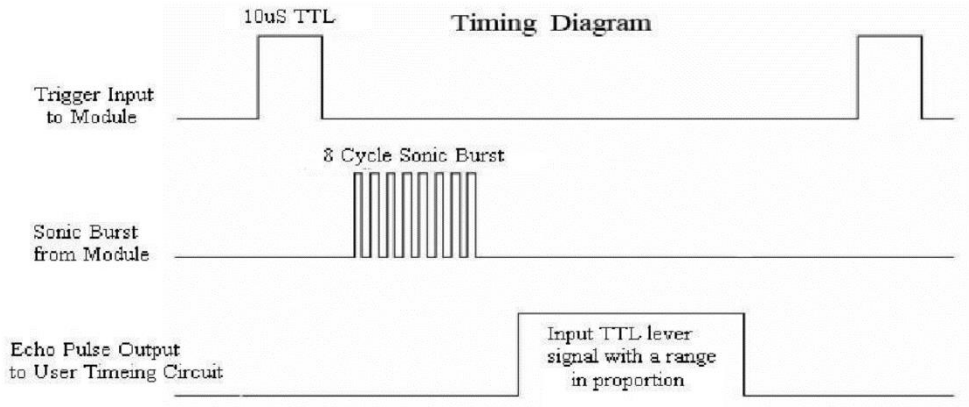
- Tajmer A0 -> povezan na LED3 i LED4
- Tajmer A1 -> povezan na HC-SR04 senzor (Echo pin)
- Tajmer A2 -> koristi se za softversko multipleksiranje displeja
- Tajmer B0 -> povezan na HC-SR04 senzor (Trig pin)
- Paralelni portovi -> povezani ka svim spomenutim eksternim periferijama
- USCI A1 -> povezan na USB konektor

- Karakteristike senzora HC-SR04

Ultrazvučni senzor HC-SR04 se satoji od ultrazvučnog predajnika i prijemnika. Takođe ima i četiri pina, dva za napajanje (V_{DD}, GND) i dva pina za komunikaciju sa mikrokontrolerom (Trig i Echo). Merenje funkcioniše tako što senzor preko svog predajnika šalje ultrazvučni signal koji putuje do prepreke i odbija se nazad i na prijemniku se detektuje reflektovani talas. Na osnovu vremena koliko je signal putovao je moguće izračunati koliko je prepreka udaljena od senzora. Senzor može da detektuje rastojanje od 2cm pa do 4m, napaja se iz naponskog izvora 5V i radi na učestanosti od 40 KHz.

Slika 1 - HC-SR04 ultrazvučni senzor

Da bi se započela akvizicija, senzoru je potrebno poslati puls logičke jedinice u trajanju od 10μs. Nakon toga senzor šalje ultrazvučni talas putem predajnika i generiše visok logički nivo na pinu Echo koji MCU treba da očitava, i kada prijemnik detektuje dolazak poslatog signala, on obara napon na Echo pinu čime se merenje završava.



Slika 2 - HC-SR04 ultrazvučni senzor način funkcionisanja (vremenski dijagrami)

Udaljenost se može izračunati prema formuli:

$$distance = \frac{1}{2}\ EchoHighLevelTime * SoundVelocity$$

Opis rada sistema

Po uključenju napajanja, sistem sam počinje akviziciju i vrši je periodično na 65ms. Rezultate prikazuje na sedmo-segmenom displeju u decimetrima. LED diode LED3 i LED4 suže da bi signalizirale koliko je prepreka blizu/daleko. One nikada nisu obe uključene. Pritiskom na taster S2 je moguće promeniti trenutno aktivnu diodu. Po podizanju sistema, najpre svetli dioda LED3.

Diode LED3 / LED4 mogu da rade u dva režima rada:

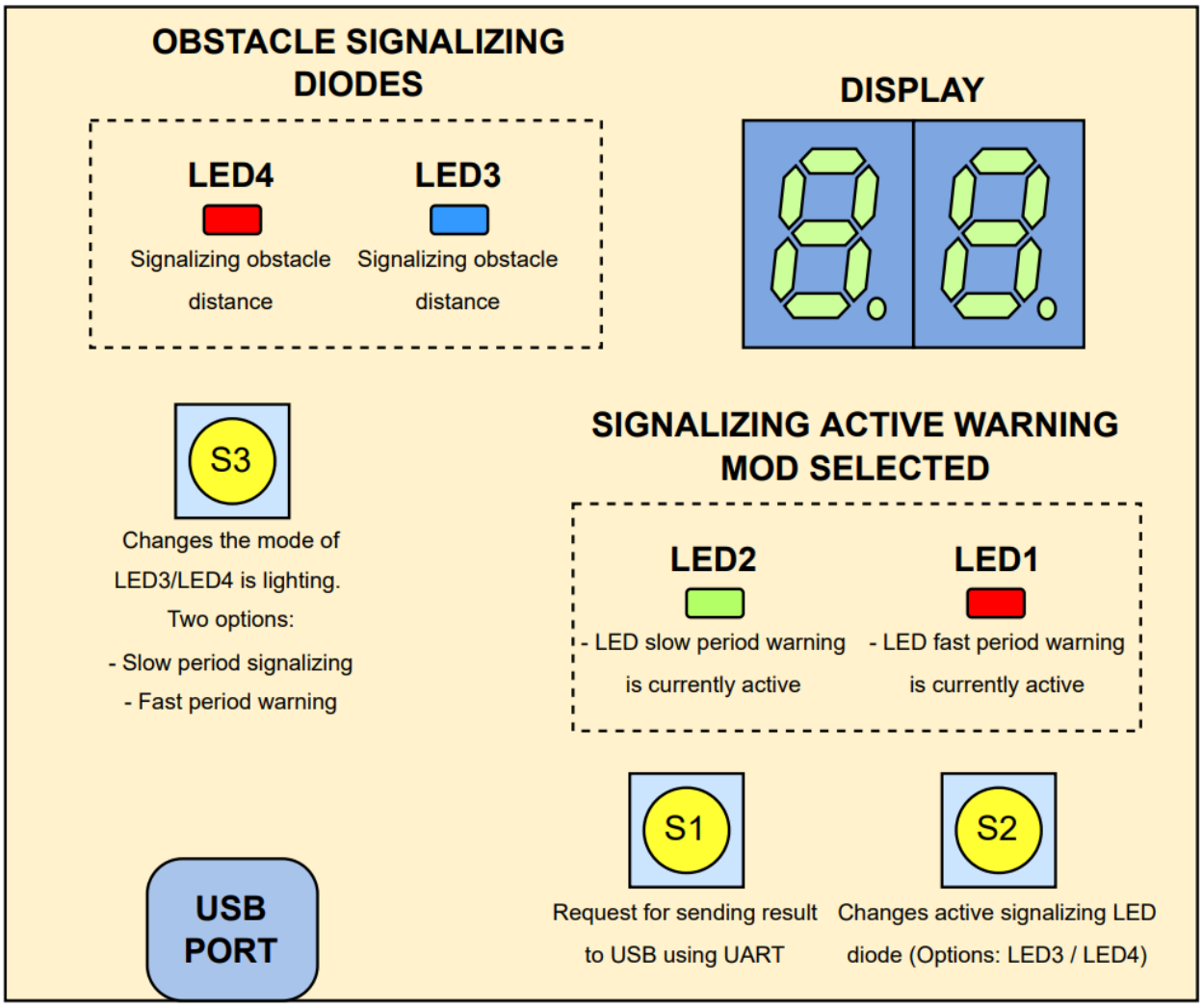
- slow warning - dioda treperi periodično svake sekunde (1s), tako da što je prepreka bliža, to je dioda u roku od 1s više vremena osvetljena, a što je prepreka dalja, to je LED dioda u toku jedne periode (1s) manje vremena osvetljena. Količina osvetljenosti u toku jedne periode srazmerna je udaljenosti prepreke (perioda je uvek 1s, a Duty Cycle je srazmeran udaljenosti).
- fast warning - dioda treperi učestalije ukoliko se prepreka približava, ili se učestanost smanjuje ako se prepeka udaljava. Dakle perioda treperenja diode je manja što je prepreka bliža, a veća što je prepreka udaljenija. U toku jedne periode, dioda je pola vremena uključena najpre, a pola vremena se izgasi (praktično perioda se menja u toku vremena, a Duty Cycle je uvek jednak trenutnoj polovini periode).

Pritiskom na taster S3 se menja režim rada. Diode LED1 i LED2 ukazuju na to koji je režim rada trenutno odabran. Ukoliko je odabran slow warning režim rada, dioda LED2 je uključena, dok je LED1 isključena i obrnuto kada je fast warning režim rada. Inicijalno po podizanju sistema, sistem počinje sa radom u slow warning režimu rada.

Pritiskom na taster S1, pokreće se proces slanja izmerenih rezultata u decimetrima preko USB interface-a ka računaru. Ta komunikacija je ostvarena pomoću USCI A1 interne periferije koja radi u UART režimu i šalje $8-\text{bitne}$ podatke brzinom $9600\ \text{bps}$, sa $2\ stop$ bita i bez bita koji služi za proveru parnosti $(9600, 8\text{N}2)$.

Pritiskom na bilo koji od pomenutih tastera (S1, S2, S3), vrši se softversko debaunsiranje pritisnutog tastera u okviru kojeg se zabranjuje prihvatanje ponovnog zahteva za prekid od strane pritisnutog tastera. Na taj način se izbegava slanje više uzastopnih zahteva za prekid od strane tastera koji je pritisnut koji bi nastali usled oskakivanja tastera i neidalnog kontakta.

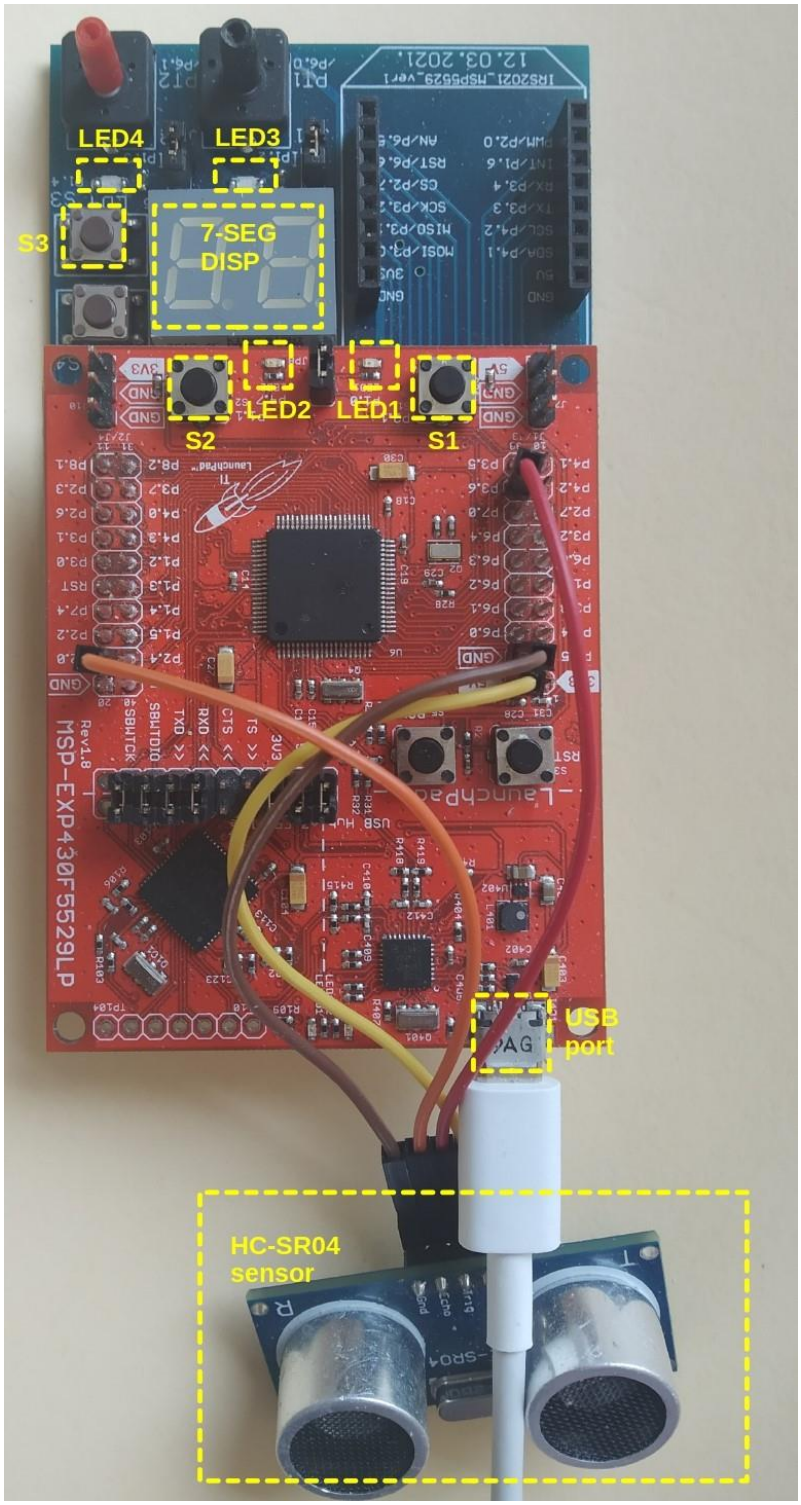
- Jednostavno korisničko uputstvo



Slika 3 - Jednostavno korisničko uputstvo

- Važne napomene

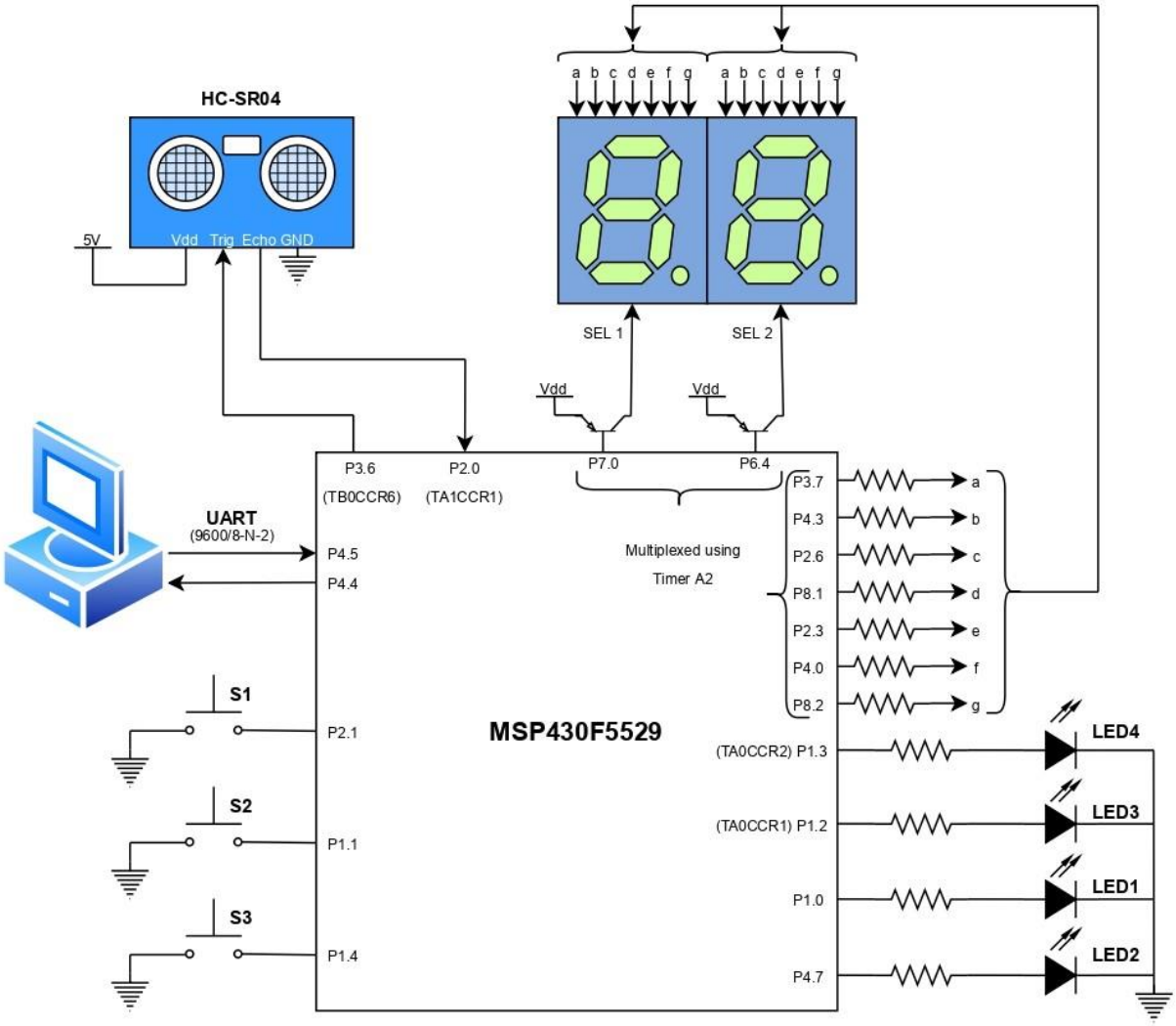
- Veze koje povezuju MCU i HC-SR04 senzor nisu najjače, tako da se dešava da sistem gubi kontakt sa senzorom u pojedinim trenutcima (položajima senzora). U tom slučaju je potrebno prstima pričvrstiti veze koje povezuju MCU i senzor kako bi kontakt bio potpun i da bi senzor mogao ispravno da komunicira sa MCU.
- Dešava se nekada da se prilikom slanja putem UART-a druga cifra izmerenog rezultata ne pošalje uvek tačna. Nije ustanovljeno do sada koji je razlog tome.
- Nije potpuno provereno ka kom stanju će iskonvergirati sistem ukoliko se dese pritisci više različitih tastera (S1, S2, S3) istovremeno. Pretpostavka je da će sistem ispravno raditi, ali nije garantovano u svim mogućim slučajevima.



Slika 4 - Slika sistema sa označenim periferijama od interesa

- Blok šema sistema

Blok šema celokupnog sistema je prikazana na slici 5 i ona opisuje povezanost delova sistema.



Slika 5 - Blok šema sistema

Arhitektura softvera

Program je implementiran u “bare metal” kodu preko jednostavnog prototipa koji podseća na Scheduler, koji dodeljuje vreme određenom tasku (koji je spreman) da se izvršava. Taskovi su prikazani na sledećoj strani na kojoj je moguće videti celokupnu arhitekturu softvera sistema i njegovu povezanost sa periferijama. Ostale programske funkcije, koje služe za početno konfigurisanje sistema, proračunavanje potrebnih parametara u toku merenja, i td. su implementirane kao posebne funkcije i izmeštene iz glavnog koda programa (u kojem se samo pozivaju). Takođe i svi bitni makroi koji se koriste u glavnom programskom kodu su izmešteni u poseban fajl.

Svi fajlovi koji pripadaju sistemu su:

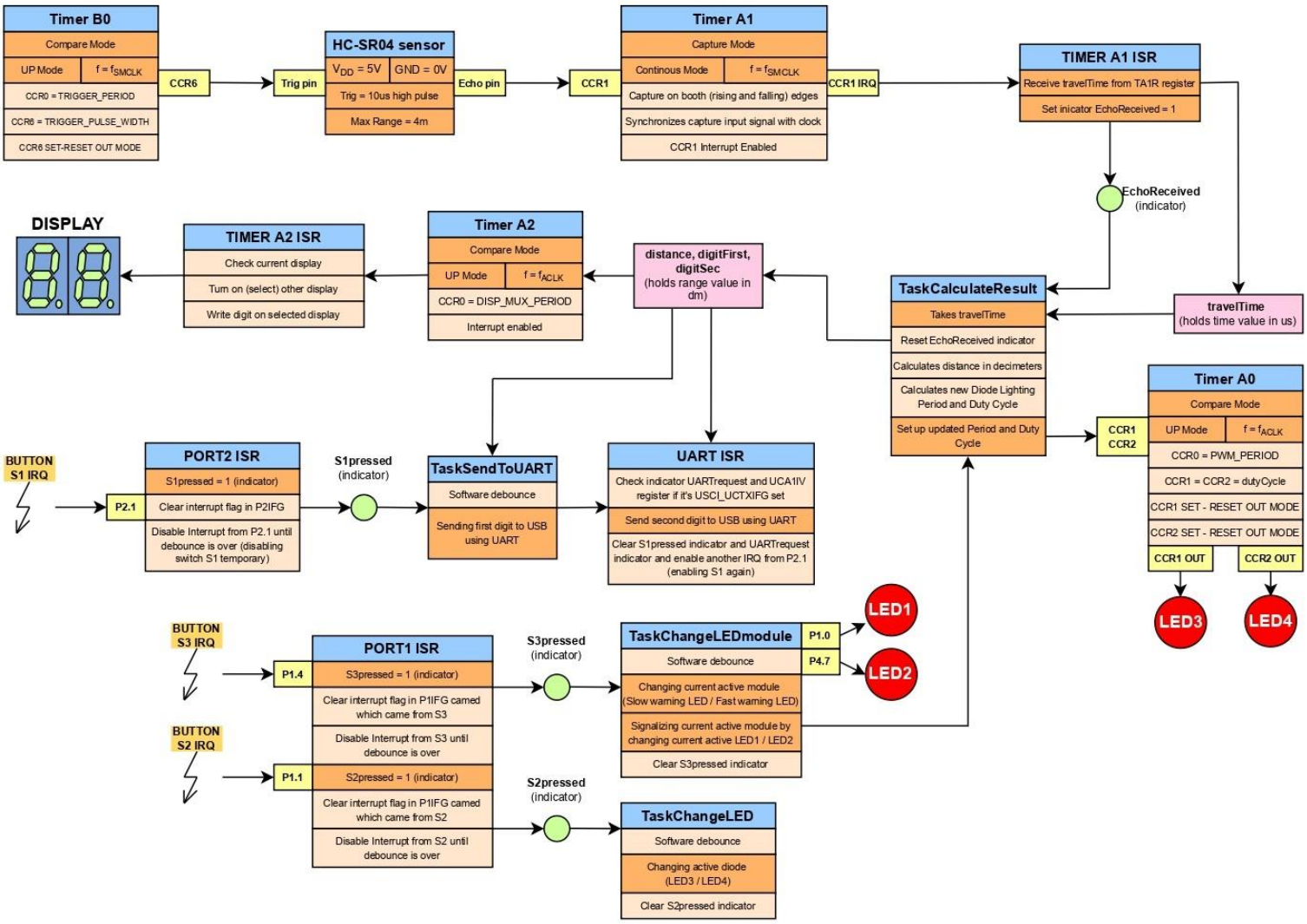
- functions.h - sadrži sve važne makroe koji se koriste u glavnom kodu (napisan je na C programskom jeziku)
- functions.c - sadrži definicije pomoćnih f-je koje se kasnije koriste u glavnom kodu (napisane na C programskom jeziku)
- main.c - kod u kome je smešten glavni kod programa (napisan na C programskom jeziku)
- p2_isr.asm - kod u kome je smešten kod prekidne rutine koji se tiče paralelnog porta 2 (napisan u asemblerskom jeziku)
- uart_isr.asm - kod u kome je smešten kod prekidne rutine koji se tiče USCI A1 interne periferije MCU-a (napisan u asemblerskom jeziku)

Kod koji je pisan u C programskom jeziku se ne prevodi često najoptimalnije (često ga je moguće realizovati znatno jednostavnije i brže izvržavanje traženih funkcionalnosti ako se kod piše u čistom asemblerskom programskom jeziku). To je posebno važno kod prekidnih rutina koje je poželjno da se što ranije izvrše, kako bi sam program mogao da što pre nastavi sa izvršavanjem glavnog programa i kako ne bi propustio neke važne događaje (koji takođe generišu prekidne zahteve). Zato su u okviru ovog projekta dve prekidne rutine realizova direktno asemblerskim jezikom, dok su ostale u C programskom jeziku. Poželjno bi bilo da su i druge prekidne rutine realizovane u asemblerskom programskom jeziku, gde bi sistem bio još optimalniji.

Periferije koje generušu zahteve za prekidom:

- prekidači S1, S2, S3 (koji su povezani na pinove paralelnih portova P1, P2)
- Tajmer A1 (koji je povezan na Echo pin senzora)
- Tajmer A2 (čija prekidna rutina se koristi za multipleksiranje aktivnog 7-segmentnog displeja)

Na slici 6 je prikazana arhitektura softvera.



Slika 6 - Arhitektura softvera sistema

Zaključak

Celokupni sistem radi ispravno, i nisu uočene veće nepravilnosti do sada. Sistem ima relativno veću potrošnju i njegove prednosti u vidu LPM režima nisu iskorišćene u ovoj projektnoj realizaciji. Među veće potrošače mogu se obrojati elementi MCU-a koji rade na višim učestanostima (1MHz), a to su CPU, tajmer B i tajmer A1. Tajmeri A0 i A2 rade na učestanosti od približno 32KHz. Softver ovog sistema u memoriji zauzima sledeći prostor:

- u RAM memoriji 184B (što je oko 2% ukupnog kapaciteta RAM memorije koja je veličine 8192B). U okviru RAM memorije, .stack sekcija zauzima 160B, dok .data sekcija zauzima 24B
- u Flash memoriji 3960B (što je oko 8% ukupnog kapaciteta Flash memorije, koja je veličine 48000B). U okviru Flash memorije, .text sekcija zauzima 3852B, .const sekcija 80B, .cinit sekcija 20B, .text_isr sekcija 8B.

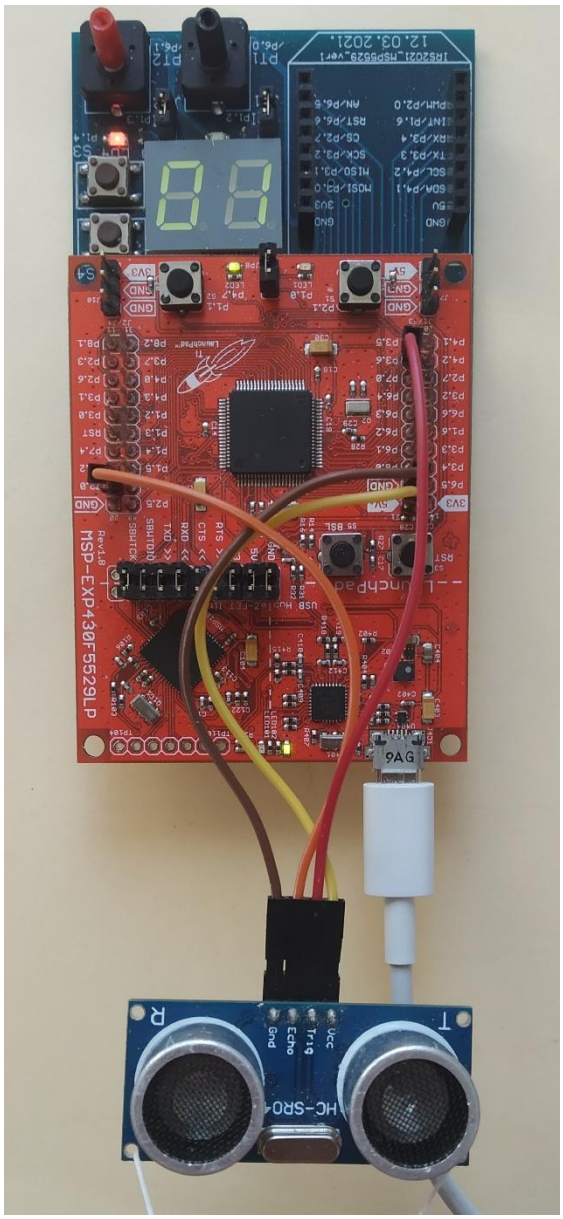
- Konačna realizacija u praksi

Takođe treba napomenuti da je sistem na slikama ispod prototip i da je za neku praktičnu primenu potrebno uraditi sledeće stavke:

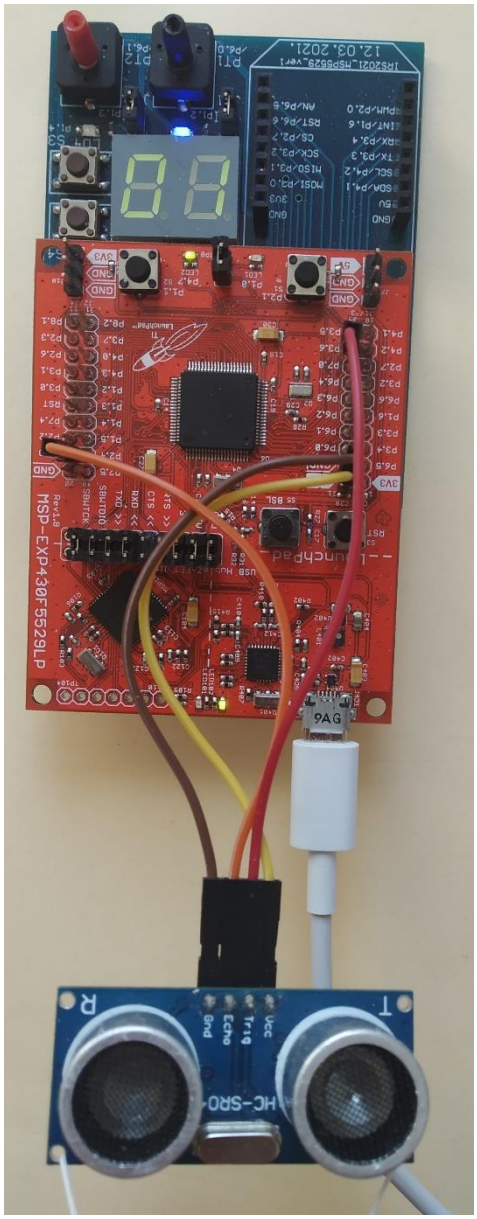
- vezice koje spajaju senzor i pinove MCU-a nisu sasvim prisne (tako da se dešava da se gubi kontakt), pa bi ih bilo potrebno zalemiti
- bilo bi poželjno da se MCU izopšti zajedno sa neophodnim periferijama i izrutira na posebnom PCB (Printed Circuit Board), gde bi zauzimao manje mesta i bio bi znatno kompaktniji
- za tastere bi se mogli uzeti boljeg kvaliteta tasteri ili prekidači (postojeći ne ostvaruju baš najbolji kontakt prilikom pritiska, posebno S3)
- sistem bi se morao upakovati u odgovarajuće kućište kako bi bio zaštićen od spoljašnjih dešavanja (zavisno gde bi se primenjivao)

- Prikaz sistema u toku rada

Na sledećim slikama je prikazan sistem u trenutku kada je ispred senzora postavljena prepreka na malo više od 1dm. Na obe slike sistem uspešno detektuje prepreku na udaljenosti većoj od 1dm i to prikazuje na displeju. Takođe, LED diode uspešno trepere (skoro da su svo vreme uključene, a mala je pauza kada su ugašene jer je sistem u slow-warning režimu rada).



Slika 7 - Prikaz sistema dok je u slow warning režimu rada i dok je uključena LED4



Slika 8 - Prikaz sistema dok je u slow warning režimu rada i dok je uključena LED3

- Moguća nadogradnja i unapređenja

Preostale dostupne periferije na platformi koje nisu upotrebljene do sada su:

- potenciometri POT1 i POT2
- taster S4
- mogućnost za dodavanjem dodatnih periferija

Predlozi kako ih je moguće upotrebiti i proširiti/unaprediti funkcionalnosti:

1) Moguće je upotrebiti ADC12 periferiju MCU-a na koju je povezan POT1 / POT2 za komunikaciju i dodatne funkcionalnosti implementirati putem nje. Na primer da ako se na POT1 očita vrednost koja je veća od polovine maksimalne, da se akvizicija vrši ručno (jednokratno - pritiskom korisnika na taster S4), a ako je ispod polovine maksimalne vrednosti, da se tada vrši akvizicija automatski (putem tajmera B kao i do sada).
2) Moguće je upotrebiti na primer POT2 da se uđe u LPM, tj. da posluži kao "power prekidač" kojim se sistem šalje u "sleep" režim rada. Kada se na primer POT2 namota na više od polovine od maksimalne vrednosti, generiše se signal koji se obrađuje u prekidnoj rutini i u glavnom programu na osnovu koga bi sistem oslobodio periferije koje nisu neophodne da se koriste i snizio učestanost takta na kome radi ceo sistem. Prelazak u "sleep" režim rada bi mogao da omogući isključenje 7-segmentnog displeja, LED diode i prestanak sa akvizicijom i trigger-ovanjem HC-SR04 senzora, smanjenje učestanosti glavnog takta CPU-a, isključiti privremeno tajmere koji nisu neophodni (potencijalno isključiti i neki ADC ako se ne koristi). Potom omogućiti da dok je sistem u takvom režimu smanjene potrošnje, namotavanjem POT2 ispod polovine maksimalne vrendosti se sistem "probudi" i nastavi sa uobičajenim radom kada se vrši merenje i prikaz rezultata (upozorenja).
3) Realizovati dodatne funkcionalnosti pomoću dodavanja opcionih periferija (koliko preostali pinovi mogu da podrže) za potrebe ispisa na npr. LCD displeju.
4) Dodavanje dodatnih senzora koji detektuju udaljenost od prepreke (na sve četiri strane) tako da bi se mogla stvarati predstava o tome da li je sistem okružen nekim preprekama oko sebe i koliko blizu. Ovakav sistem bi se mogao upotrebiti u spreza sa još nekoliko ovakvih mikrokotrolera i senzora za upravljanje jednostavnijim vozilom.
5) Računati i ispisivati ubrzanje/brzinu kretanja sistema na displeju ukoliko se sistem kreće i okružen je nepokretnim preprekama. Na osnovu nekoliko prethodno izmerenih rezultata udaljenosti od prepreke bi bilo moguće računati brzinu kretanja sistema i ubrzanje.
6) Slanje rezultata putem Bluetooth (ili nkeog drugog RF) predajnika ka nekom drugom uređaju.
7) Prihvatanje informacija putem Bluetooth (ili nkeog drugog RF) prijemnika kojima bi bilo moguće konfigurisati sistem (pokretati akviziciju, birati načine rada i druge funkcionalnosti koje takođe moraju biti podržane softverski od strane MCU-a).
8) Pisanje više delova koda u asemblerskom jeziku na osnovu čega bi se programski kod izvršavao znatno brže i imao bi bolji odziv na događaje na koje reaguje. Posebno kod prekidnih rutina bi to bilo poželjno učiniti.

Literatura

[1]https://www.ti.com/lit/ug/slau208q/slau208q.pdf?ts=1649203301292&ref_url=https%253A%252F%252Fwww.google.com%252F

[2]https://www.ti.com/lit/ds/symlink/msp430f5529.pdf?ts=1649164280276&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FMSP430F5529

[3] https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

[4] Nenad Jovičić, Integrisani računaski sistemi - materijali sa predavanja

[5] Integrisani računaski sistemi - materijali sa vežbi

[6] https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/

[7] https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

[8] Haris Turkmanović, Sistemi u realnom vremenu - Vežbe 2021/2022

[9] Ivan Popović, Sistemi u realnom vremenu - Slajdovi sa predavanja (http://tnt.etf.bg.ac.rs/~oe4srv/index_files/Page323.html)