Univerzitet u Beogradu Elektrotehnički fakultet

Integrisani Računarski Sistemi

Projekat – Ispis trenutne pozicije na LCD displeju uz pomoć MSP430 mikrokontrolera sa GPS Ublox Neo-6m modulom

Jovana Savić 2013/0243

1. Opis projekta

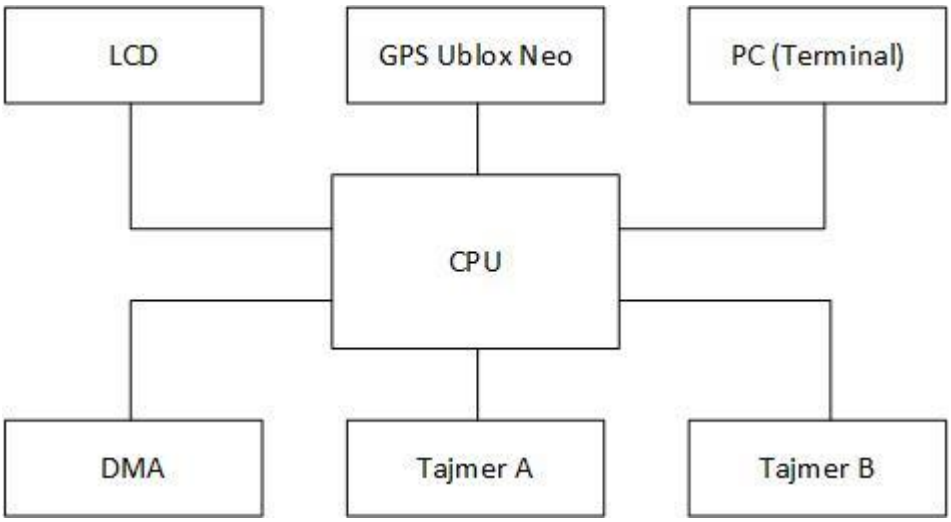
Potrebno je uz pomoć mikrokontrolera MSP430F5438A na LCD displeju ispisati geografsku širinu i dužinu kao i *dilution of precision* koji se dobijaju od GPS Ublox Neo-6m click pločice.

2. Uvedene pretpostavke, pristup problemu, korišćen hardver

Pošto se traženi podaci ne mogu odjednom ispisati na LCD displeju, uvedeno je da se na jednom ekranu ispisuju geografska širina i dužina, a na drugom *dilution of precision*. Pritiskom tastera S4 se menja ekran koji je trenutno aktivan.

Obzirom da je MSP430 mikrokontroler koji je bogat periferijama, projekat je urađen tako da se ova činjenica maksimalno iskoristi, nakon inicijalizacije hardvera na početku programa, sve dalje obrade se dešavaju u sklopu prekidnih rutina. Da bi mogla da se izvede ovakva organizacija projekta napravljena je mašina stanja koja dekoduje poruku u samoj prekidnoj rutini. Prenos tačno primljene poruke je urađen uz pomoć DMA kontrolera, a ispis na LCD uz pomoć tajmera A i B. Pošto je projekat testiran na istom mestu (pa se samim tim i uvek primaju iste poruke) uvedeno je da se uz pomoć terminala preko UART-a takođe šalju poruke. Ovo omogućava bolje testiranje projekta.

Na slici 2.1. je prikazan hardver koji je korišćen.



Slika 2.1. Korišćen hardver

Komunikacija sa GPS pločicom se obavlja preko UART-a. Pločica je povezana sa portom ***P5.7*** – ***UCA1RX***.

Komunikacija sa računarom se takođe obavlja preko UART-a – ***UCA0RX*** na portu ***P3.5***.

LCD displej je povezan sa portom ***P8***. Njemu se šalju 4-bitni podaci i ***EN*** i ***RS*** signali.

DMA kontroler se nalazi u samom kućištu kao i tajmeri. Koriste se kanali 0 i 1 za prenos podataka.

3. GPS Ublox Neo-6m

Podaci se šalju serijski preko UART-a. Postoje opcije korišćenja 9600 i 4800 *baud rate*-a. Izabran baud rate je 9600 (default vrednost).

GPS šalje nekoliko poruke definisane NMEA standardom[1] sa učestanošću od oko 5Hz. Izabrano je da se parsiraju GPGGA i GPGSA poruke (format prikazan na slikama 3.1. i 3.2.) i iz njih čitaju potrebni prodaci. Na kraju svake poruke se šalje "*" praćena *checksum*-om. Vrednost *checksum*-e u poruke se dobija rađenjem operacije eksluzivnog ili njenog sadržaja. Ova vrednost se koristi da se proveri da li je pristigla poruka ispravna – tokom prijema se računa *check sum*-a i ukoliko se izračunata vrednost poklapa sa poslatom poruka je najverovatnije ispravno stigla.

GGA - essential fix data which provide 3D location and accuracy data.

```
 $GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47

Where:
     GGA          Global Positioning System Fix Data
     123519       Fix taken at 12:35:19 UTC
     4807.038,N   Latitude 48 deg 07.038' N
     01131.000,E  Longitude 11 deg 31.000' E
     1            Fix quality: 0 = invalid
                               1 = GPS fix (SPS)
                               2 = DGPS fix
                               3 = PPS fix
                               4 = Real Time Kinematic
                               5 = Float RTK
                               6 = estimated (dead reckoning) (2.3 feature)
                               7 = Manual input mode
                               8 = Simulation mode
     08           Number of satellites being tracked
     0.9          Horizontal dilution of position
     545.4,M      Altitude, Meters, above mean sea level
     46.9,M       Height of geoid (mean sea level) above WGS84
                      ellipsoid
     (empty field) time in seconds since last DGPS update
     (empty field) DGPS station ID number
     *47          the checksum data, always begins with *
```

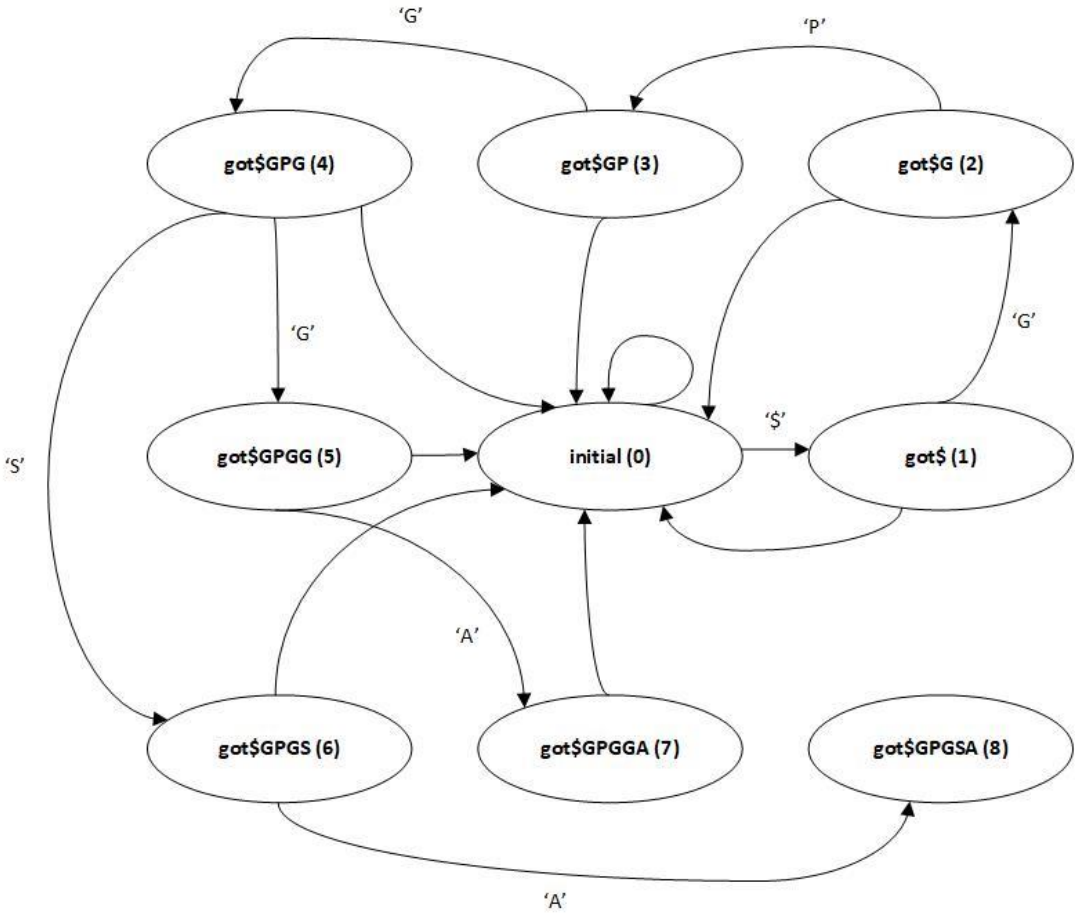
Slika 3.1. GGA format poruke

Parsiranje poruka se izvršava u prekidnoj rutini na osnovu mašine stanja date na slikama 3.3, 3.4. i 3.5. Mašina stanja je kodovana tako da se prelazak u sledeće stanje prevodi u inkrementiranje ili duplo inkrementiranje vrednosti trenutnog stanja. Ovo je urađeno je se ove operacije svode na sabiranje sa vrednostima koje se mogu generisati uz pomoć generatora konstanti, čime se štedi jedan ciklus čitanja iz memorije. Izlaz odgovarajućeg stanja se u asembleru elgantno prevodi u dodavanje vrednosti trenutnog stanja pomnožene sa dva na PC i zatim skok na odgovarajući deo koda.

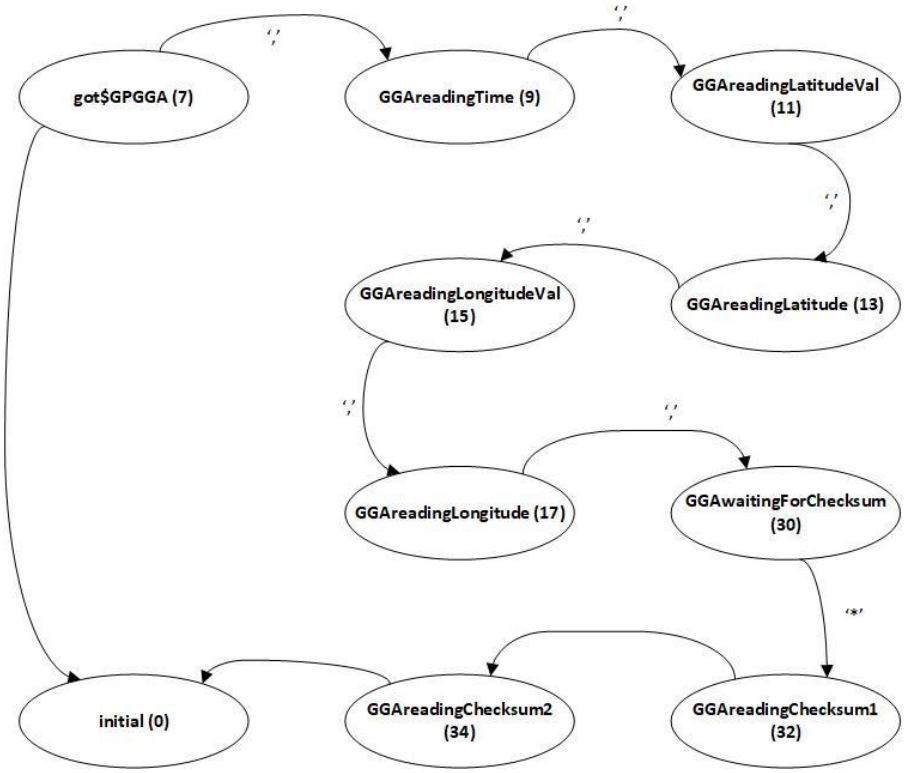
[1] http://www.gpsinformation.org/dale/nmea.htm

```
  $GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39

Where:
     GSA      Satellite status
     A        Auto selection of 2D or 3D fix (M = manual)
     3        3D fix - values include: 1 = no fix
                                       2 = 2D fix
                                       3 = 3D fix
     04,05... PRNs of satellites used for fix (space for 12)
     2.5      PDOP (dilution of precision)
     1.3      Horizontal dilution of precision (HDOP)
     2.1      Vertical dilution of precision (VDOP)
     *39      the checksum data, always begins with *

  $GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39
```

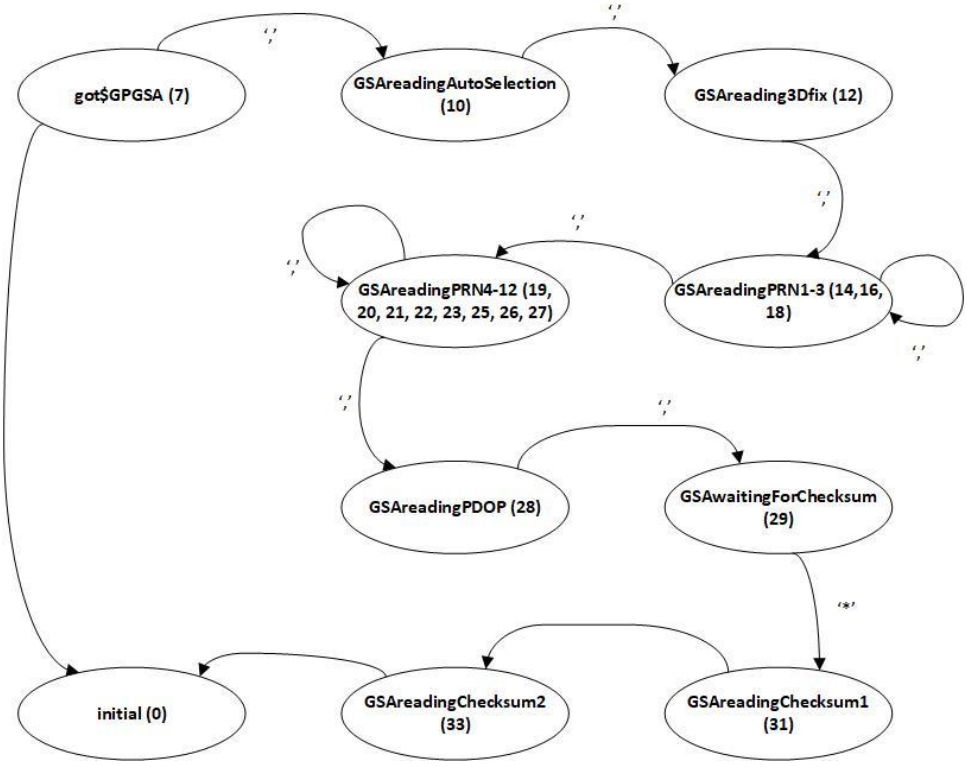
Slika 3.2. GSA format poruke



Slika 3.3. Parsiranje koda poruke – mašina stanja



Slika 3.4. Parsiranje koda poruke – GGA poruka



Slika 3.5. Parsiranje koda poruke – GSA poruka

Pisanje prekidne rutine u asembleru omogućava da se prilično komplikovano parsiranje poruke izvršava jako brzo. Da je prekidna rutina pisana u C-u, morao bi da postoji niz pokazivača na funkcije za odgovarajuća stanja poruke i tada bi se stanje koristilo za indeksiranje poziva odgovarajuće funkcije, tj. izlaza mašine stanja. Iako logički praktično isto, ovo rešenje zahteva pozivnje funkcije umesto skoka, što znači da se automatski gubi vreme prilikom promene konteksta.

Primljene poruke se smeštaju u dva bafera – GSAmessage_tmp, GGAmessage_tmp. Prilikom prijema karaktera se računa *checksum*-a. Ukoliko je izračunata suma jednaka poslatoj, RX prekid se zabranjuje da ne bi došlo do korupcije podataka i startuje se prenos preko odgovarajućeg kanala DMA kontrolera. DMA prenosi podatke iz bafera u kome se nalazi primljena poruka u bafer u kome se nalaze trenutna pozicija, odnosno *dilution of precision*.

Inicijalizacija UART-a je data u fajlovima ***uart.h*** i ***uart.c***, a prekidna rutina se nalazi u fajlu ***gps_irs_dma.asm***.

4. LCD displej

Za korišćenje LCD displeja je napravljena hardverska apstrakcija čiji interfejs sadrži funkcije za ispis karaktera, pozicioniranje kursora, ispis stringa sa brisanjem displeja pre ispisa, ispis stringa počev od zadate pozicije i brisanje displeja. Posebno, napravljena je funkcija kojom se LCD displej inicijalizuje.

Pomoćne funkcije koje nisu predviđene da budu deo interfejsa su funkcija kojom se šalje komanda, funkcija kojom se na *data* linije postavlja *nibble* (ova funkcija ne vodi računa o tome da li je dati nibble deo podatka ili komande) i funkcija koja obezbeđuje odgovarajući *enable* signal.

Date funkcije su deklarisane u fajlu ***lcd.h***, a definisane u fajlu ***lcd.c***. Osim ovih funkcija, u fajlu ***lcd.h*** su definisani i makroi za najčešće komande.

5. DMA kontroler

Kako je RX prekid zabranjen tokom preuzimanja ispravno primljene poruke, poželjno je da se dati prenos izvrši što brže. Zbog toga je za prenos korišćen DMA kontroler, a ne procesor. Napravljene su funkcije kojima se inicijalizuju kanali 0 i 1 i po jedan kanal se koristi za prenos jedne vrste poruke. Podaci su bajtovski i šalju se blokovski sa inkrementiranjem. Postavljeno je da DMA prekidom javi završetak prenosa.

U UART prekidnoj rutini se po prijemu validne poruke zabranjuje RX prekid i startuje DMA prenos na odgovarajućem kanalu. Kada završi prenos bloka, u DMA prekidnoj rutini se ponovo dozvoljava DMA prenos (ali ne inicijalizuje), i ponovo se dozvoljava UART RX prekid. Takođe se brojač tajmera A postavlja na 0 tako da se u slučaju da je dozvoljen prekid tajmera A (odnosno u toku je ispis) iznova radi ispis.

Funkcije za inicijalizaciju DMA kontrolera su deklarisane u ***dma.h*** fajlu, a definisane u ***dma.c*** fajlu. Prekidna rutina je napisana u asmebleru, nalazi se u ***dma_irs.asm*** fajlu.

6. Tajmer A

Tajmer A je korišćen za ispis na LCD displeju. Koristi ACLK izvor takta (32768 Hz), a u registru CCR0 se nalazi vrednost 1. Tajmer radi u UP modu, i generiše prekid kada izbroji do vrednosti 1 – dakle na 1/32768 s. U prekidnoj rutini ovog tajmera se u zavisnosti od toga koji je trenutno ekran aktivan ispisuje jedan karakter iz bafera sa porukama – GSAmessage, GGA message. U vezi sa ovim tajmerom je i promenljiva ucTimerA0counter koja vodi računa o tome koji se karakter trenutno ispisuje (i samim tim da li je potrebno preći u novi red). Kada završi sa prenosom svih karaktera koje je potrebno ispisati na trenutno aktivnom ekranu, u prekidnoj rutini se zabranjuju dalji prekidi ovog tajmera.

Tajmer se inicijalizuje uz pomoć funkcije definisane u fajlu ***config.h***, a deklarisane u fajlu ***config.c***. Prekidna rutina je data u fajlu ***timerA0_isr.asm***.

7. Tajmer B

U jednoj o prethodnih verzija je ispis vršen svaki put kada DMA završi prenos, odnosno DMA je dozvoljavao prekid tajmera A. Međutim, u opštem slučaju ovo može da bude "overkill" rešenje. Ukoliko se položaj sporo menja, to povlači sa sobom osvežavanje ispisa na LCD displeju veliki broj puta za jednu istu poruku. Da bi se dala sloboda da se osvežavanje vrši ređe kada za tako nešto ima smisla, uveden je tajmer B kojim se na svakih približno 5s osvežava LCD displej, tako što se na 5s dozvoljava prekid tajmera A. U ovom slučaju nema potrebe da DMA vodi računa o ispisu, odnosno dozvoljava prekid tajmera A u svojoj prekidnoj rutini, već se poruke praktično poliraju na svakih 5s. Naravno, ostavljeno je prostora da se vrednost osvežavanja menja.

Inicijalizacija se nalazi u fajlovima ***config.h*** i ***config.c***, a prekidna rutina u fajlu ***timerB0_isr.asm***.

8. Taster

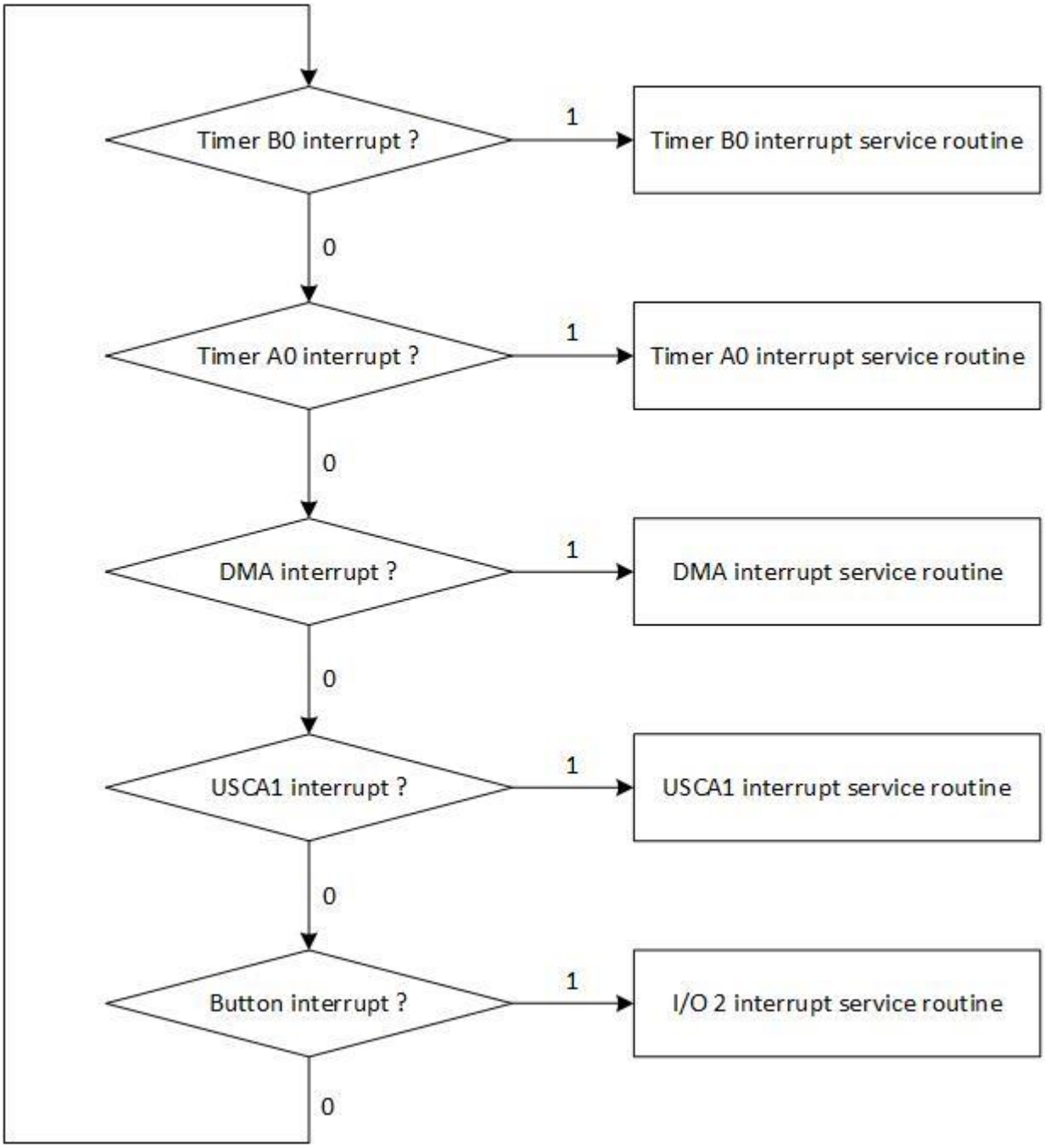
Korišćen je taster S4 koji se nalazi na portu P2.7. Inicijalizacija koja obuhvata postavljanje smera porta i dozvolu prekidna na silaznu ivicu se nalazi u fajlovima config.h i config.c. Kada se detektuje silazna ivica generiše se prekid. U prekidnoj rutini se provodi određeno vreme i nakon toga ponovo proverava da li je signal i dalje na aktivnoj vrednosti. Ukoliko jeste, smatra se da je taster pritisnut i menja se aktivan ekran. Nakon toga se i dozvoljava prekid tajmera A i resetuje njegov brojač. Na ovaj način se omogućava promena ispisa.

Inicijalizacija se nalazi u fajlovima ***config.h*** i ***config.c***, a prekidna rutina u fajlu ***button.asm***.

9. Algoritam programa

Program je napravljen tako da bude "event driven" - izvršavanje programa je posledica prijema karaktera preko UART-a, isteka određenog vremenskog perioda i pritiska tastera. Na osnovu prioriteta prekida je dat algoritam po kome se izvršava dati program.

Treba imati na umu da je u situaciji u kojoj se čeka na DMA prekid USCA1 prekid zabranjen i obrnuto. Naravno, prekidi su po ulasku u neku od prekidnih rutina zabranjeni.



Slika 9.1. Algoritam programa

10. Zaključak

Predloženo rešenje ima određene prednosti. Osnovna prednost je u tome što je korišćen DMA kontroler za prenos podataka iz privremenih bafera u bafere u kojima se čuvaju konačni podaci. Pošto je neophodno proveravati validnost poruke čuvanje trenutno primljene poruke i poslednje tačno primljene poruke ne može da se izbegne. Isto tako, u opštem slučaju ne bi trebalo dozvoliti RX prekid prilikom prenosa iz privremenih u konačne bafere jer u suprotnom može doći do korupcije podataka. Rešenje uz pomoć DMA kontrolera je bio "najbezbolniji" način da se ovo reši. Može se primetiti da je prenos iz privremenih bafera u one u kojima se nalaze poruke mnogo brži nego učestanost sa kojom se šalju poruke, pa bi se moglo ukazati na to da se dok se poslednja dobra primljena poruka prenese neće pojaviti nova iste vrste. Zato možda i može da se izbegne zabrana RX prekida. Međutim, to ne bi stvorilo toliku uštedu u vremenu, a ostavlja prostora za greške. To bi moglo da se zameri u slučaju da je brzina kretanja velika, pa gubitak poruke dovodi do velike nepreciznosti, ali treba napomenuti da su koordinate koje korišćeni GPS šalje već u krajnjim granicama dozvoljene preciznosti (greška je oko 100m), pa bi ova GPS pločica u takvom slučaju ionako bila neadekvatna.

Druga prednost ovog rešenja je u mašini stanja koja je korišćena za parsiranje poruke. Skoro sav rad potreban za parsiranje je iz vremenskog domena prebačen u domen koda programa, što za program ovako male veličine nije toliki problem ako se ne računa preglednost i elegantnost koje su usput izgubljene. Naravno, parsiranje drugih prouka bi zahtevalo više stanja i preglednost koda sa svakom novom porukom bi postajala eksponencijalno gora. Zbog toga je uvedeno da se parsiraju samo dve poruke iako se isti podaci mogu pročitati i iz drugih vrsta poruka. Ukoliko bi bilo bitno da sistem bude robustan više izvora istih podataka bi moglo da se iskoristi za dodatnu proveru dobijenih koordinata (iako *checsum*-a smanjuje verovatnoću greške, ona ipak nije nula – pogotovo jer postoji veliki broj zareza). Naravno, smisao ovakvog pristupa nestaje kako broj podataka koje je potrebno preuzeti sa GPS pločice raste.

Najveća mana ovog rešenja jeste ispis celog karaktera uz pomoć tajmera – na ovaj način se program dosta dugo zadržava u prekidnoj rutini. U nekoj narednoj verziji bi to prvo trebalo ispraviti i pobrinuti se da ova prekidna rutina traje kraće.

Drugi veliki nedostatak ovog rešenja je u tome što je debaunsiranje tastera urađeno isto u prekidnoj rutini, dakle opet se dosta vremena provodi u prekidnoj rutini bez potrebe. Isto tako, u nekoj narednoj verziji, ovo bi trebalo uraditi uz pomoć tajmera.

Predloženo rešenje je daleko od optimalnog, ali su dosta dobro iskorišćene prednosti koje nudi MSP430. Takođe, gde god je bilo prostora rešenje je urađeno tako da nadogradnja bude što lakša, odnosno da rešenje bude što je više moguće generalizovano. Ipak, tamo gde je trebalo voditi računa o brzini – prekidne rutine i mašina stanja korišćena za parsiranje poruka, dato je veoma usko i specifično rešenje, obzirom da je brzina izvršavanja ipak bila primarni cilj.

10. Literatura

1. http://tnt.etf.bg.ac.rs/~oe4irs

2. http://www.ti.com/lit/ds/symlink/msp430f5438a.pdf

3. http://www.ti.com/lit/ug/slau208q/slau208q.pdf

4. http://processors.wiki.ti.com

5. http://tnt.etf.rs/~oe4irs/old/RS_MSP430F5438A_sch.pdf

6. http://www.gpsinformation.org/dale/nmea.htm

7. https://circuitdigest.com/article/16x2-lcd-display-module-pinout-datasheet

8. https://www.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf