Univerzitet u Beogradu
Elektrotehnički fakultet

Alen Baković, 3052/2012

GPS/GSM lokator

projekat iz predmeta: 32-bitni mikrokontroleri i primena

profesor:
dr Dragan Vasiljević

Beograd, februar 2013.

Sadržaj

Glava 1

Uvod

U ovom dokumentu je opisan način na koji je realizovan projekat iz predmeta *32-bitni mikrokontroleri i primena*, koji se drži na master studijama katedre za elektroniku na Elektrotehničkom fakultetu u Beogradu. U izveštaju će biti opisan hardver koji se koristio prilikom izrade projekta kao i softver. Biće prikazani rezultati, odnosno način na koji funkcioniše napravljeni uređaj.

Cilj projekta bio je upoznavanje studenata sa 32-bitnom ARM arhitekturom. U ovom konkretnom slučaju korišćen je mikrokontroler baziran na ARM Cortex M3 familiji mikroprocesora.

Glava 2

Projektni zadatak

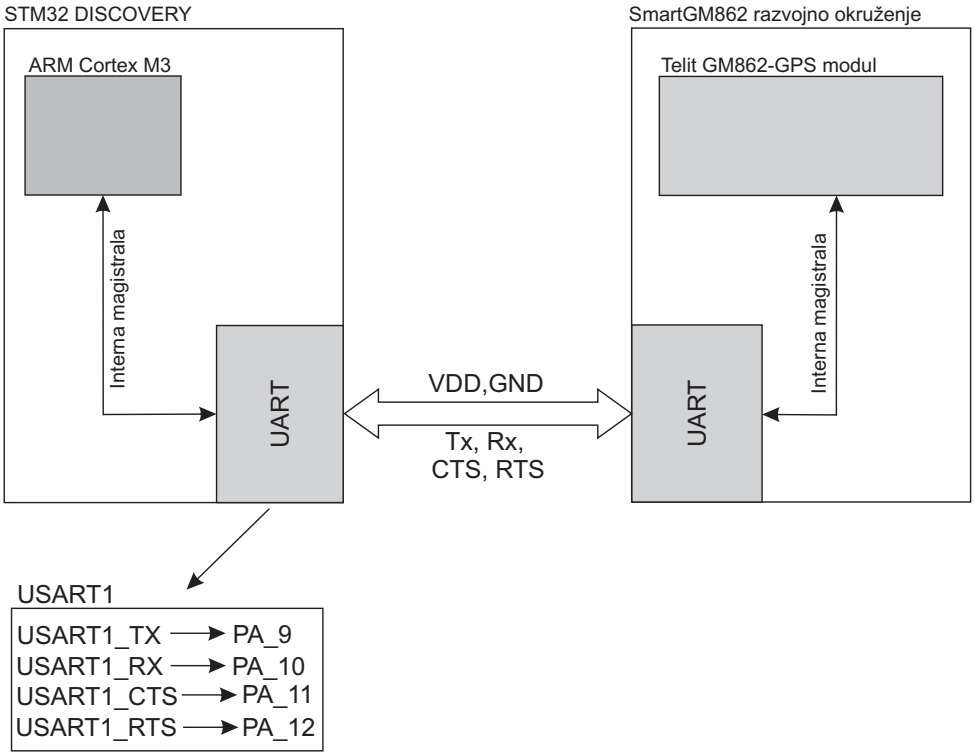
Pomoću Telit GM862-GPS modula i STM32 Discovery razvojnog okruženja realizovati GPS/GSM lokator.

Očekivana funkcionalnost uređaja: Korisnik poziva uređaj na kome se nalazi SIM kartica. Uređaj odbija poziv, zatim u narednih nekoliko sekundi šalje poruku na broj sa kojeg je došao poziv koja sadrži koordinate gde se uređaj nalazi. Ukoliko uređaj nije u stanju da vrati informaciju o poziciji, o tome obaveštava korisnika takođe putem SMS poruke.

Glava 3

Realizacije projekta

Za realizaciju projekta korišćena su dva razvojna okruženja. Na jednom razvojnom okruženju (STM32 DISCOVERY) nalazi se ARM Cortex M3 32-bitni mikrokontroler, dok se na drugom razvojnom okruženju (SMART GM862) nalazi Telit modul (Telit GM862-GPS). Ova dva modula su između sebe povezana preko UART-a. Blok šema sistema data je na slici 3.1.



Slika 3.1: *Blok dijagram sistema.*

3.1 GPS-Global Positioning System

Na razvojnom okruženju SMART GM862 nalazi se Telit GM862-GPS modul. On se koristi za komunikacuju preko GSM mreže između uređaja i korisnika i za dobijanje informacije o poziciji korišćenjem GPS-a.

GPS projekat je razvijen sedamdesetih godina 20. veka od strane Ministarsva odbrane SAD. Zasnovan je na satelitima koji kruže u Zemljinoj orbiti i pruža informacije o lokaciji i vremenu. Ima 24 satelita koja se kreću oko Zemlje. U počeku je bio korišćen samo u vojne svrhe. Danas je korišćenje GPS-a veoma rasprostranjeno, koristi se u vojne i civilne svrhe. Ovaj sistem mogu koristiti svi koji imaju GPS prijemnik. Ukupna greška određivanja pozicije je oko 15 metara.

3.2 Resursi mikrokontrolera

U pogledu hardverskih resursa mikrokontrolera ovaj sitem je veoma jednostavan. U realizaciju GPS/GSM lokatora korišćeni su USART1, TIMER2 i NVIC.

3.2.1 USART

STM32 DISCOVERY ima 3 USART-a, u ovom slučaju korišćen je USART1 čiji su pinovi na STM32 DISCOVERY ploči obeleženi sa PA9, PA10, PA11, PA12 (slika 3.1). SMART GM862 razvojno okruženje ima translator nivoa tako da je pored četiri osnovne linije koje se koriste za USART potrebno povezati VDD i GND koji se nalaze na STM32 DISCOVERY na SMART GM862. UART se koristi za komunikaciju između Telit modula i ARM Cortex-a M3. U daljem tekstu je prikazano na koji način je konfigurisan UART. Prvo je prikazana konfiguracija samih pinova na razvojnom okruženju jer njihova primarna funkcija je GPIO, a zatim i konfiguracija USART-a.

```
/*******************************************************
Configuration of STM32 pins to their alternate function
*******************************************************/
void configure_uart_pins(void){

  GPIO_InitTypeDef GPIO_InitStructure;

  /* Configure USART1 Tx as alternate function push-pull */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  /* Configure USART1 Rx as input floating */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  /* Configure USART1 CTS as input floating */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_11;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
  GPIO_Init(GPIOA, &GPIO_InitStructure);
```

```
  /* Configure USART1 RTS as alternate function push-pull */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_12;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
  GPIO_Init(GPIOA, &GPIO_InitStructure);
}
```

```
/**********************
Configuration of USART1
**********************/
void configure_uart(void){

  USART_InitTypeDef USART_InitStructure;

  USART_InitStructure.USART_BaudRate = 9600;
  USART_InitStructure.USART_WordLength = USART_WordLength_8b;
  USART_InitStructure.USART_StopBits = USART_StopBits_1;
  USART_InitStructure.USART_Parity = USART_Parity_No;
  USART_InitStructure.USART_HardwareFlowControl =
                                    USART_HardwareFlowControl_RTS_CTS;
  USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
  USART_Init(USART1, &USART_InitStructure);

  /* Enabling USART1 interrupts */
  USART_ITConfig(USART1, USART_IT_RXNE, ENABLE);
}
```

3.2.2 TIMER2

U realizaciji projekta korišćen je TIMER2. Tajmer je podešen tako da generiše prekid na svakih 100ms i u prekidnoj rutini inkrementira brojač. Prekidna rutina je trivijalna pa neće biti dat njen kod. Funkcija kojom je izvršeno podešavanje tajmera prikazana je ispod.

```
/*****************************************************
Configuration of timer2 to generate periodical interrupts
*****************************************************/
void configure_timer2(void){

 TIM_TimeBaseInitTypeDef TIM_TimeBaseInitStruct;
 /* Time base configuration */
 TIM_TimeBaseInitStruct.TIM_Period = 270;
 TIM_TimeBaseInitStruct.TIM_Prescaler = 8880;
 TIM_TimeBaseInitStruct.TIM_ClockDivision = 0;
 TIM_TimeBaseInitStruct.TIM_CounterMode = TIM_CounterMode_Up;

 TIM_TimeBaseInit(TIM2, &TIM_TimeBaseInitStruct);

 /* Enabling timer2 interrupts */
 TIM_ITConfig(TIM2,TIM_IT_Update, ENABLE);
}
```

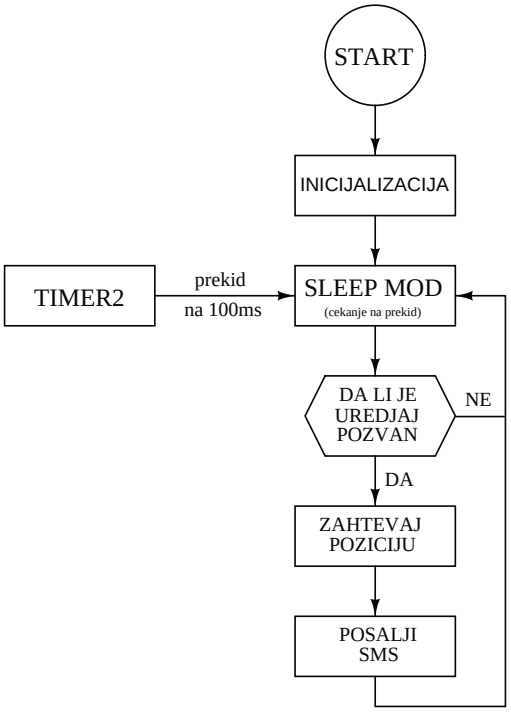
3.2.3 NVIC-Nested Vectored Interrupt Controller

NVIC je treća periferija koja se koristi. Ova periferija upravlja svim prekidima, maskira ih, odredjuje prioritet... Za realizaciju projekta bilo je potrebno opsluživati prekide od dve periferije USART-a i TIMER-a. NVIC je tako podešen da je prekid koji generiše USART većeg prioriteta od prekida koji generiće TIMER2, i ukoliko se u prekidnoj rutini TIMER-a javi prekid USART-a, prekidna rutina TIMER-a će se prekinuti i krenuće sa izvršavanjem prekidna rutina USART-a. Funkcija u kojoj je izvršeno podešavanje NVIC-a data je ispod.

```
/*************************************************************
  Setting the NVIC to enable USART1 and TIMER2 interrupts
  *************************************************************/
void configure_NVIC(void){

  NVIC_InitTypeDef NVIC_InitStructure;

    /* Enable the USART1 Interrupt */
  NVIC_InitStructure.NVIC_IRQChannel = USART1_IRQn;
  NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
  NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1;
  NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
  NVIC_Init(&NVIC_InitStructure);

     /* Enable the TIMER2 Interrupt */
  NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQn;
  NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0xF;
  NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0xF;
  NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
  NVIC_Init(&NVIC_InitStructure);
}
```

3.3 Glavi program

Izvorni kod glavnog programa se nalazi u fajlu *main.c*. Ovde neće biti prikazan ceo glavni program, biće izostavljene funkcije za inicijalizaciju periferija, Telit modula. Biće prikazan deo od kada se uđe u beskonačnu petlju. Algoritam rada programa prikazan je na slici 3.2. Nakon početne inicijalizacije periferija procesor ulazi u mod spavanja odnosno mod smanjene potrošnje. Ovo je izvedeno korišćenjem WFI instrukcije. WFI instrukcija uvodi procesor u SLEEP mod, a iz njega izlazi nakon što dođe zahtev za prekidom. Iz tog razloga je potreban tajmer koji je podešen tako da na svakih 100 ms generiše prekid.



Slika 3.2: *Algoritam rada glavnog programa.*

Nakon što je tajmer generisao prekid, procesor ulazi u prekidnu rutinu i izvršava je, a nakon toga kreće sa daljim izvršavanjem programa. Proverava se da li neko od korisnika poziva uređaj. Ukoliko je to slučaj, glavni program iz poruke koju prima preko UART-a od Telit modula čuva broj koji ga poziva i na koji treba da pošalje poruku. Odbija poziv kao znak da je poziv primljen i da korisnik treba da očekuju SMS poruku. Zatim šalje Telit modulu komandu da mu ovaj vrati informaciju o lokaciji tj koordinate. Ukoliko Telit modul vrati ispravno koordinate poslaće se SMS poruka sa istima. Ukoliko Telit modul nije uspeo da uspostavi vezu sa satelitima i nije poslao koordinate, korisniku će se poslati poruka da pozicija ne može biti izračunata. Nakon što je poruka poslata procesor upet ulazi u SLEEP mod i čeka da neko od korisnika pozove pa da opet uđe u petlju.

Prilikom komuniciranja sa Telit modulom i slanja SMS poruka, mora se pričekati neko vreme pa da se nastavi sa izvršavanjem programa. Kako bi se izbegle takozvane „mrvte“ petlje gde procesor radi NOP instrukciju ili nešto slično realizovana je funkcija **wait_ms**. Ovoj instrukciji se prosleđuje parametar koji predstavlja koliko milisekundi treba čekati. Ukoliko je potrebno sačekati 400 ms, ispravna sintaksa za pozivanje funkcije je *wait_ ms(400).* Kod funkcije je dat ispod. tim2_int je parametar koji se inkrementira pri svakom ulasku u prekidnu rutinu tajmera 2.

```
/*****************************************
Wait function is using WFI instruction;
Input parametar time should be given in ms
*****************************************/
extern int tim2_int;
void wait_ms(int time){

  time=(time/100) + 1;
  tim2_int=0;

  while(tim2_int<time){
    __WFI();
   }
}
```

3.3.1 Format poruke

Format poruke koju vraća uređaj korisniku koji je pozvao uređaj je sledeći:

DDs MM.MMMMm N/S
DDDs MM.MMMM E/W

DD predstavlja stepene.
MM.MMMM su minuti u decimalnom zapisu.
N/S (North/South) - Sever/Jug.
E/W (East/West) - Istok/Zapad.

Na primer, ukoliko se od uređaja dobije poruka:
44s 50.6978m N
020s 29.6091m E
To znači da je pozicija uređaja 44 stepena, 50.6978 minuta severno. 20 stepeni, 29.6091 minuta istočno.

U izveštaju neće biti poseban deo o testiranju rada uređaja jer je testiranje prlično trvijalno. Uređaj je potrebno povezati i ubaciti SIM karticu. Testiranje je vršeno tako što je pozivan broj SIM kartice koja je ubačena u uređaj. Kao odgovor dobija se jedna od moguće dve poruke. Prva poruka sadrži koordinate u formatu koji je prikazan iznad. Ukoliko Telit modul nije uspostavio vezu sa satelitima korisniku će biti odgovoreno SMS porukom sa sadržajem:
„Trenutno nije dostupna informacija o poziciji. Pokušajte kasnije.“

Glava 4

Zaključak

U izveštaju je ukratko opisan način na koji je realizovan projekat. Za detaljniji uvid treba pogledati izvorne fajlove.

Ono što treba napomenuti, a tiče se funkcionisanja uređaja je to da će uređaj raditi ispravno samo ukoliko je broj sa kojeg se poziva uređaj sedmocifren. Npr 064 7777777. Ukoliko je broj šestocifren uređaj neće raditi. Ovaj deo nije teško popraviti, ali nije implementiran zbog nemogućnosti da se testira jer nijedan šestocifreni broj nije bio na raspolaganju.

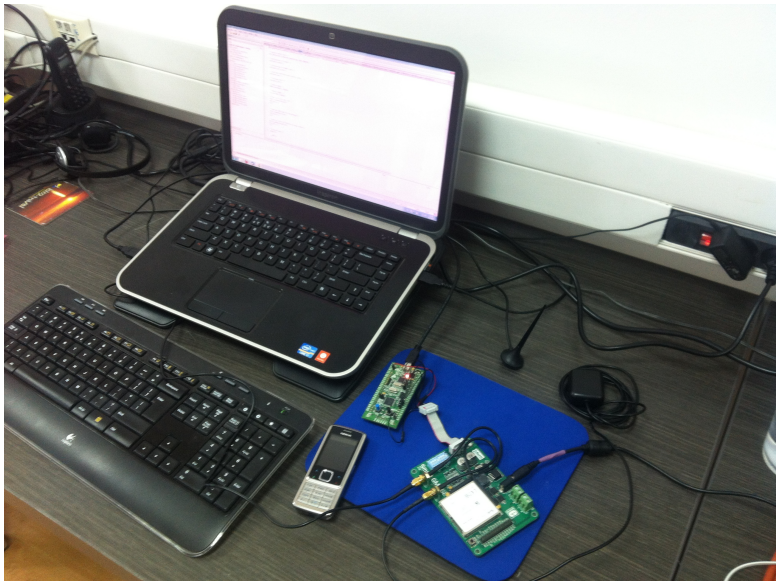
S'obzirom da je program tako napravljen da je procesor u SLEEP modu i da se čeka na prekid koji dolazi na 100 ms, a potom se vrši obrada ukoliko ima potrebe, procesor nije opterećen i malo troši. Ukoliko se desi da niko ne poziva uređaj nekoliko minuta ili duže, što je realna situacija, procesor će trošiti vrlo malo struje i neće biti opterećen.

Uređaj može izvršavati još neku aplikaciju ukoliko je to potrebno. Ako se ne bi prepravljao kod i ostavila *wait_ ms()* funkcija onako kako je napisana, korišćenjem WFI instrukcije, sva dodatna obrada mogla bi se raditi u prekidnoj rutini. Kada se pozove WFI instrukcija procesor ulazi u mod smanjene potrošnje i čeka na prekid koji će ga izvući iz tog režima. Kada se primi zahtev za prekidom, procesor ulazi u prekidnu rutinu, izvršava je nastavlja da izvršava kod koji se nalazi posle WFI instrukcije. Ukoliko je na primer potrebno da se periodično očitavaju naponi može se iskoristiti prekidna rutina nekog tajmera opšte namene. Tajmer bi trebalo podesiti da periodično generiše prekid. Ukoliko je procesor u SLEEP modu po dolasku zahteva za prekid on bi se probudio, ušao u prekidnu rutiu očitao napone, smestio rezultate u memoriju i ukoliko nije prošlo dovoljno vremena koje je definisano kao parametar u funkciji *wait_ ms* opet bi se vratio u SLEEP mod.

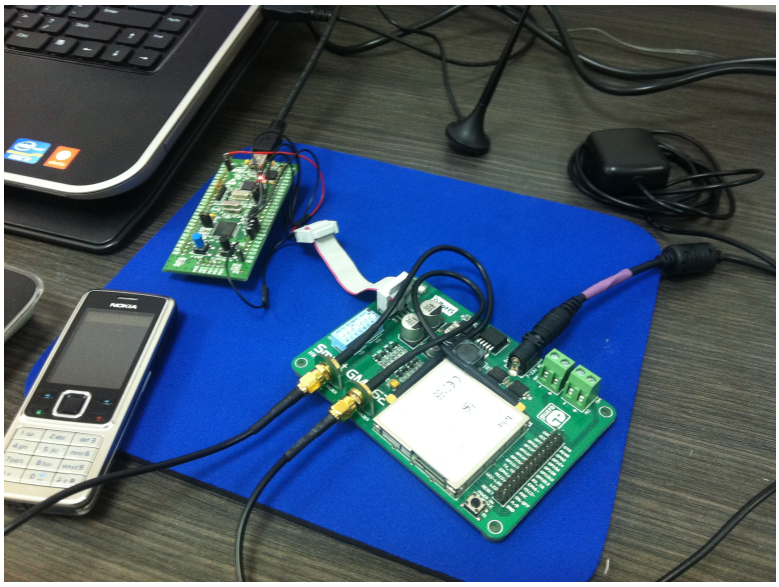
Ukoliko bi bilo potrebno da se u glavnom programu ne čeka u SLEEP modu već samo da se proveri da li je prošlo određeno vreme, pa ukoliko jeste odraditi šta je potrebno i nastaviti dalje, a ukoliko nije onda preskočiti taj deo, raditi dalje pa u sledećoj iteraciji opet proveriti da li je prošlo dovoljno vremena, to bi zahtevalo veoma sitnu prepravku koda. Ne bi se više koristila funkcija *wait_ ms()*. Umesto nje samo bi trebalo proveravati flag koji se u ovom konkretnom slučaju inkrementira u prekidnoj rutini tajmera 2 na svakih 100 ms. Prvo taj fleg treba postaviti na 0, zatim proveriti da li je on dostigao određenu vrednost (koju vrednost zavisi od toga koliko vremena treba da prođe). Ukoliko je fleg dostigao željenu vrednost (inkrementiran dovoljan broj puta, odnosno prošlo je dovoljno vremena) treba izvršiti šta je potrebno. Ukoliko fleg nije dostigao željenu vrednost onda treba krenuti dalje. Po završenom ciklusu beskonačne *while* petlje opet se kreće iz početka. Ponovo će doći na red proveravanje flega i u zavisnosti od toga preduzimaće se akcije.

Na ovaj način je obezbeđeno da se u glavnom programu ne čeka u SLEEP modu, međutim sada je procesor mnogo zauzetiji nego kada se koristi WFI instrukcija jer više ne ulazi u mod niske potrošnje nego neprestano radi u beskonačnoj *while* petlji.

Slike kako sistem izgleda kada se sve poveže (STM 32 Discovery, Telit, PC) prikazane su ispod.



Slika 4.1: *Izgled sistema*



Slika 4.2: *Izgled sistema*

Bibliografija

[1] *Predavanja i vežbe iz predmeta 32-bitni mikrokontorleri i primena*

[2] *Joseph Yiu, The definitive guide to the ARM Cortex M3*

[3] *Telit AT Commands Reference Guide*

[4] *STM32F100xx reference manual*

[5] *Wikipedia–the free encyclopedia*

[6] *Literatura sa interneta*