SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 3329

SUSTAV ZA BEŽIČNO MJERENJE KUTNE AKCELERACIJE I BRZINE

Ivan Križanić

Zagreb, lipanj 2013.

Sadržaj

Uvod

Suvremeni čovjek je okružen ugradbenim računalnim sustavima i koristi ih za obavljanje vrlo različitih svakodnevnih problema. Pritom takvi sustavi obavljaju funkcije koje čovjek iz nekog razloga ili ne može ili ne želi obavljati i u tome leži njihova važnost.

Poseban značaj imaju uređaji za bežično komuniciranje jer smanjuju značajne troškove provođenja žičane mreže između uređaja, a vrlo važni su i uređaji za mjerenje različitih vanjskih podražaja koristeći senzore.

Brojanje koraka je aktivnost koju svaki čovjek može obavljati, ali se kroz neko vrijeme potkradaju pogreške jer mozak podsvjesno potiskuje koncentraciju s aktivnosti koju obavlja automatski.

U tu je svrhu zamišljen automatski bežični sustav koji bi kroz određeno kontinuirano vrijeme mjerio broj koraka koje osoba prijeđe. Koristio bi se senzorski sustav sastavljen od troosnog akcelerometra za određivanje akceleracije noge korisnika u svakom trenutku. Prenosio bi te podatke bežično do točke u kojoj bi se oni mogli obraditi i prikazati. Obradba podataka bi odredila u kojem trenutku korisnikova noga miruje i u tom trenutku definirala početak koraka. Svako sljedeće mirovanje noge značilo bi jedan prijeđeni korak.

U navedenu svrhu koristit će se bežični komunikacijski sustav *eZ430-Chronos* koji se sastoji od *Chronos* sata i pristupne točke. *Chronos* sat će se koristiti za obradbu i prikaz podataka, a pristupna točka će služiti kao senzor pričvršćen na korisnikovu nogu.

1. Struktura sustava

Sustav za mjerenje kutne akceleracije i brzine ostvaren je koristeći razvojni alat *Texas Instrumentsa*, *eZ430-Chronos*, i senzor kutne akceleracije i brzine.

1.1. Značajke sustava

eZ430-Chronos[1][2] je u potpunosti reprogramabilno razvojno okružje temeljeno na *CC430F6137* mikrokontroleru u ručnom satu i mikrokontroleru temeljenom na 8051 arhitekturi unutar pristupne točke.

Radi se o *ultra-low power* sustavima koji troše vrlo malo električne energije i time omogućavaju što dulje trajanje baterije. Ovisno o načinu upotrebe uređaja, životni vijek baterije se može značajno skratiti.

Način rada	**Prosječna jakost struje**	**Trajanje baterije**
Početni zaslon	8.9 μA	28 mjeseci
Prikaz vremena i datuma	9.0 μA	27.7 mjeseci
Kontinuirano mjerenje temperature	10.0 μA	25.0 mjeseci
Kontinuirano mjerenje nadmorske visine	18.0 μA	13.8 mjeseci
Kontinuirano mjerenje akceleracije	166.0 μA	1.5 mjeseci
Kontinuirano primanje podataka *BlueRobin* protokolom	40.0 μA	6.2 mjeseci
Kontinuiran prijenos podataka *SimpliciTI* PPT protokolom	10.0 μA	25 mjeseci

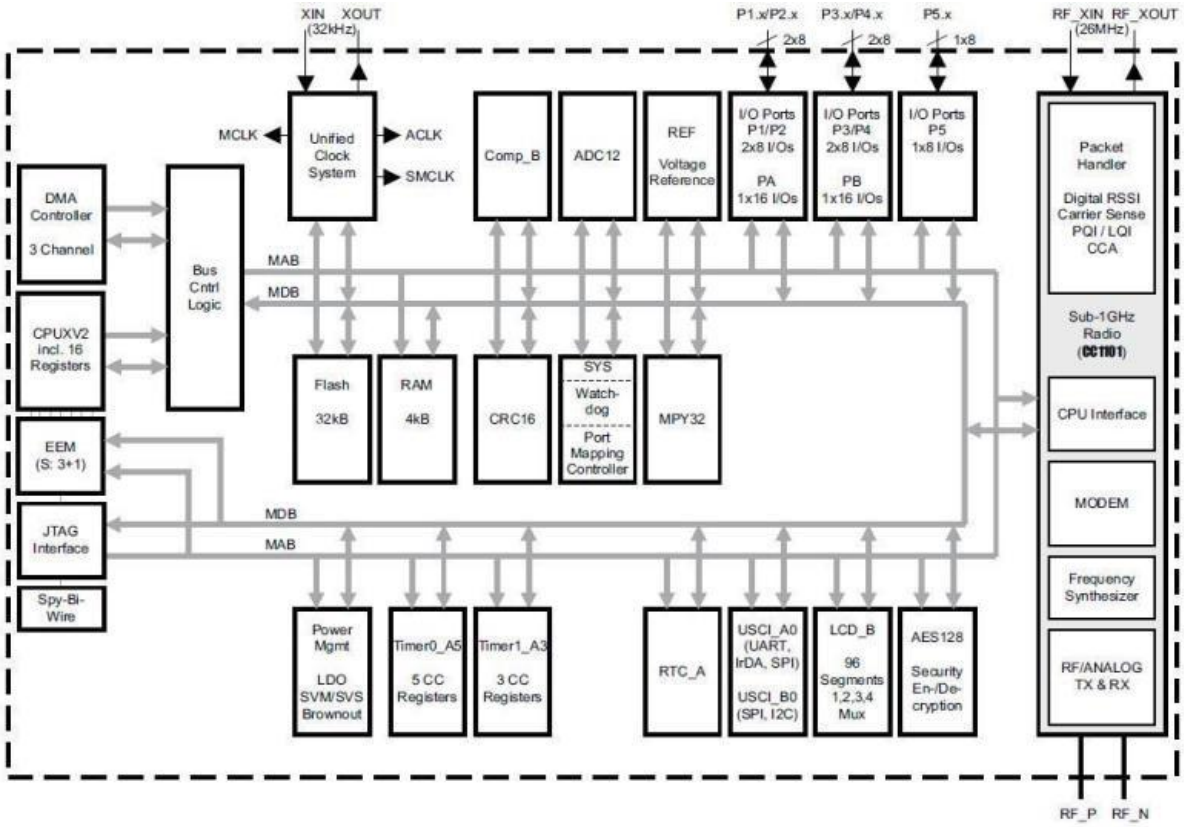
Kontinuiran prijenos podataka *SimpliciTI* SYNC protokolom	0.9 mA	8 dana
Kontinuiran prijenos podataka *SimpliciTI* ACC protokolom	3.7 mA	2 dana
Svakodnevni jednosatni prijenos podataka *BlueRobin* protokolom	10.3 μA	24.2 mjeseci
Svakodnevni jednosatni prijenos podataka *SimpliciTI* PPT protokolom	9.1 μA	25.4 mjeseci
Svakodnevni jednosatni prijenos podataka *SimpliciTI* SYNC protokolom	46.1 μA	5.4 mjeseci
Svakodnevni jednosatni prijenos podataka *SimpliciTI* ACC protokolom	169.9 μA	1.4 mjeseci

Tablica 1. Usporedba potrošnje u različitim načinima rada

U tablici 1. je dana usporedba potrošnje električne energije ovisno o načinu rada sata. Ručni sat kao napajanje koristi bateriju od 3 V, dok se pristupna točka napaja izravno s računala koristeći USB vezu.

2. eZ430-Chronos

Razvojni alat *eZ430-Chronos* je integrirani bežični sustav koji omogućava stvaranje bežičnih aplikacija za pametni sat. Uz odgovarajuće modifikacije originalnog uređaja i njegova kôda, ova platforma može obavljati brojne funkcije. Neke od njih su: bežično upravljanje otvaranjem i zatvaranjem elektroničke brave, upravljanje kretanjem robotskog automobila ovisno o nagibu sata, odbrojavanje, štoperica, kontrola temperature prostorije i bežična sklopka za svjetiljku.



Slika 1. Struktura mikrokontrolera CC430F6137

Sustav se temelji na *CC430F6137* mikrokontroleru iz porodice *CC430* procesora smanjene potrošnje energije (*ultra-low-power microcontroller*) i integriranoj radiofrekvencijskoj primopredajnoj jezgri na jednoj *System-on-Chip* (*SoC*) tiskanoj pločici. Slika 1. prikazuje strukturu mikrokontrolera *CC430F6137*. Velik broj

perifernih sklopova omogućava široku primjenu mikrokontrolera iz ove porodice u vrlo različitim aplikacijama. Arhitektura *CC430* mikrokontrolera je optimizirana da bi se postiglo što dulje trajanje baterije tijekom bežičnih mjerenja vanjskih veličina i radiofrekvencijskog prijenosa. Mikrokontroler se odlikuje snažnom *MSP430* procesorskom jezgrom koja obrađuje 16-bitne podatke, koristi 16-bitne registre i generatore konstanti da bi se postigla maksimalna efikasnost koda.

Serija *CC430F61xx* mikrokontrolera obuhvaća *System-on-Chip* konfiguracije *CC1111*[3] primopredajnika i *MSP430 CPU XV2* procesorske jezgre. Procesorska jezgra sadrži velik broj vanjskih jedinica, uključujući 32 kB promjenljive programske memorije, do 4 kB RAM, dva 16-bitna brojača, 12-bitni AD pretvornik s osam ulaza, integrirani temperaturni senzor i mjerač napona baterije, komparatore, podršku za USB, sklopovsko množilo, sklop za izravni pristup memoriji (DMA), sat stvarnog vremena sa alarmom, driver za LCD i do 44 ulazno-izlazna priključka.

Tipična primjena ovih mikrokontrolera uključuje bežične analogne i digitalne senzorske sustave, termostate i pametne bežične mreže.

Slika 2. Prikaz vanjskog sklopovlja na eZ430 Chronos satu

Slika 2. prikazuje izgled sklopa *Chronos* sata van zaštitnog kućišta. Prijenos podataka obavlja se na frekvencijama nižim od 1 GHz, a *eZ430-Chronos* dolazi u tri različite varijante ovisno o frekvenciji radijskog prijenosa. Radi se o frekvenciji od 433 MHz, koja je slobodna za korištenje u većini država svijeta, frekvenciji od 868 MHz u Europi i Indiji i frekvenciji od 915 MHz za tržište Sjeverne i Južne Amerike. U ovom se završnom radu radilo s *eZ430-Chronos-868*, odnosno, radiofrekvencijski prijenos se obavlja na 868 Mhz.

Chronos sat sadrži i druge sklopove kojih nema na pristupnoj točki. Radi se o 96-segmentnom LCD zaslonu s mogućnošću ispisa svih slova engleske abecede i svih brojeva, kao i nekih posebnih znakova. U sat su integrirani senzor tlaka i akcelerometar koji mjeri akceleraciju po sve tri osi. U slučaju korištenog uređaja riječ je o sklopovima *VTI SCP1000* i *VTI CMA3000* koji se više ne proizvode i zamijenjeni su u novim varijantama uređaja.

Sat dolazi programiran i za mjerenje otkucaja srca i brzine kretanja kad je spojen na senzor otkucaja srca u pojasu koji se stavlja na prsa korisnika. Pojas ne dolazi uz sat već se može posebno naručiti od njemačke tvrtke *BM innovations*.

Programiranje, prevođenje i prijenos programa na mikrokontroler vrši se koristeći dva različita razvojna okruženja. Originalni program za *Chronos* sat može se programirati koristeći *Code Composer Studio* ili *IAR Embedded Workbench KickStart*, dok se originalni program za pristupnu točku može programirati samo koristeći *IAR Embedded Workbench KickStart*. Uz promjene projektnih datoteka programe je moguće programirati i prevoditi koristeći i druge alate.

Sav radiofrekvencijski prijenos podataka vrši se s istim parametrima. Pritom je brzina prijenosa 115200 simbola po sekundi, a svaka poruka je duljine 8 bita s jednim stop bitom i bez paritetnog bita.

Slika 3. Pristupna točka CC1111

Pristupna točka *CC1111* je *System-on-Chip* primopredajnik s integriranim USB kontrolerom koji omogućava brz i lagan prijenos podataka između računala i RF pristupne točke. Dolazi s 32 kB flash programske memorije i 4 kB RAM memorije. Slika 3. prikazuje pristupnu točku *CC1111*. Smještena je u vrlo maleno kućište što omogućava velik broj različitih primjena. Pristupnu točku pokreće mikrokontroler smanjene potrošnje temeljen na arhitekturi 8051.

Radio veza je vrlo velike osjetljivosti, do oko 100 dBm pri brzini prijenosa od 1200 simbola po sekundi s maksimalnom brzinom prijenosa i do 500 tisuća simbola po sekundi. Garantira se izlazna snaga od barem 10 dBm pri svim podržanim frekvencijama.

Pristupna točka je također dizajnirana u *low power* načinu rada i ima vrlo malenu potrošnju energije, ovisno o načinu prijenosa podataka. Uz izlaznu snagu od 6 dBm primanje podataka pri 1200 simbola po sekundi troši 16.2 mA struje, a odašiljanje oko 15.2 mA.

Sklopom protiču puno manje struje u načinima rada smanjene potrošnje energije, 0.3 μA u načinu rada kada samo vanjski prekidi bude pristupnu točku i 0.5 μA u načinu rada kada vanjski prekidi i prekidi brojača uzrokuju buđenje pristupne točke.

Pristupna točka podržava širok raspon napona napajanja, od 2.0 V do 3.6 V.

3. Programiranje sustava eZ430-Chronos

Obzirom da se sustav sastoji od dvije zasebne jedinice, potrebno je posebno programirati *Chronos* sat i posebno programirati pristupnu točku.

Za programiranje *Chronos* sata se koristi USB sučelje za programiranje, koje dolazi u paketu sa *Chronos* sustavom. USB sučelje se direktno spaja na *Chronos* sat preko 4 priključka. Slika 4. prikazuje spajanje programatora na *Chronos* sat.

Slika 4. Primjer spajanja *Chronos* sata na USB programator

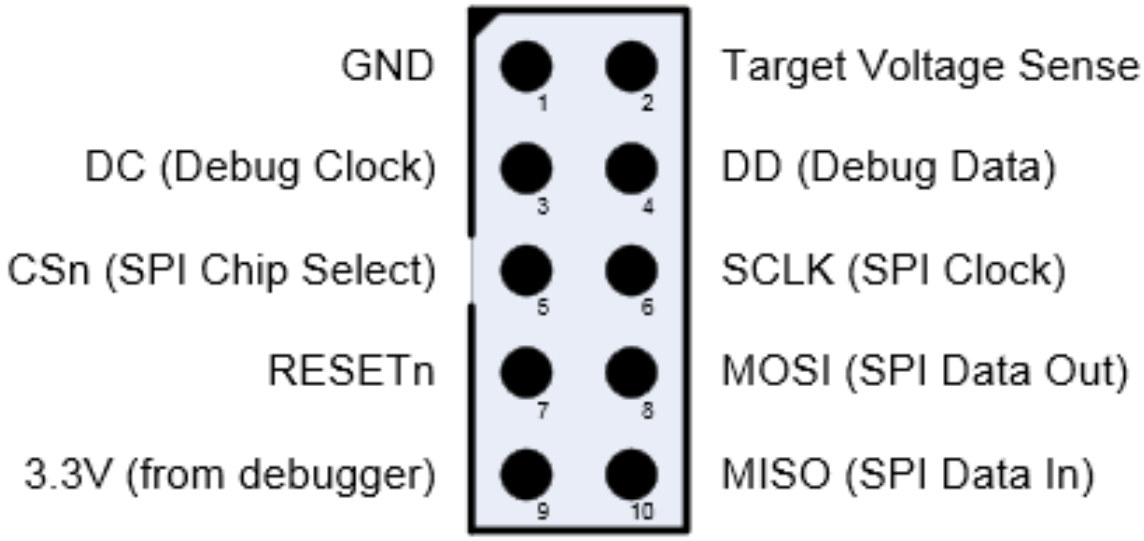
Programski sustav *Code Composer Studio* i *IAR Embedded Workbench* služe za ispravno prevođenje programa i ispravljanje pogrešaka u programu. Nakon što se programator s *Chronos* satom spoji u računalo i željeni program prođe prevođenje, *Code Composer Studio* ili *IAR* preko USB veze prenose program na *Chronos* sat. Prije programiranja sata nužno ga je odspojiti s napajanja.

Programiranje pristupne točke je nešto složenije. Iako se pristupna točka direktno spaja na računalo USB vezom nije moguće programirati pristupnu točku direktno preko USB veze. Za to se koristi *CC Debugger*[4][5]. Riječ je o sustavu koji služi za prevođenje i programiranje svih radiofrekvencijskih primopredajnika iz porodice *CCxxxx* 8051-temeljenih *System-on-Chip* uređaja proizvedenih u Texas

Instrumentsu. Radi se o *CC1110*, *CC1111*, *CC2430*, *CC2510*, *CC2511*, *CC2530*, *CC2531*, *CC2533*, *CC2540*, *CC2541*, *CC2543*, *CC2544* i *CC2545*.

Uz navedeni hardver, za programiranje su potrebni programi *SmartRF™ Flash Programmer* i *IAR Embedded Workbench* 8051. Uređaji spojeni na debugger mogu biti direktno kontrolirani kroz program *SmartRF™ Studio*.

CC Debugger se 10-pinskim kabelom spaja na odgovarajuću *CCxxxx* pločicu preko konektora koji dolazi s uređajem. Napaja se s 3.3 V napona, minimalni napon za rad iznosi 1.2 V, a maksimalni je 3.6 V.



Slika 5. Priključci CC Debuggera

Tiskana pločica *CC1111* ima samo šest vidljivih pinova na sebi i oni se koriste prilikom prevođenja programa na pristupnu točku. Da bi se izvelo prevođenje, potrebno je zalemiti priključke tiskane pločice na *CC Debugger*. Slika 5. prikazuje priključke *CC Debuggera*.

Nakon što je *CC1111* pločica nalemljena na *Debugger* može se početi s programiranjem pristupne točke kroz *SmartRF™ Flash Programmer* program. Naime, prevođenje programa kroz *IAR Embedded Workbench* 8051 rezultirat će u .hex datoteci koja će se isprogramirati na pristupnu točku jednom kad je veza s *CC Debuggerom* uspostavljena.

Nakon uspješnog programiranja potrebno je odlemiti sve veze i testirati novi program.

Slika 6. Spajanje CC Debuggera na pristupnu točku

4. Načini rada Chronos sata

Chronos sat dolazi prethodno programiran u *Sports Watch* način rada u kojem sat ima sve funkcije sportskog sata. To znači da mjeri vrijeme, datum i temperaturu, a koristeći senzor tlaka određuje nadmorsku visinu. Program ima funkcionalnost štoperice, a posjeduje i mogućnost ispisa akceleracije po sve tri osi koristeći troosni akcelerometar integriran u sat. Osim toga, sat posjeduje četiri različita načina povezivanja s pristupnom točkom.

Sva četiri načina rada predstavljaju jednosmjernu komunikaciju, u kojoj jedan čvor odašilje podatke, a drugi šalje potvrdu o primitku podataka.

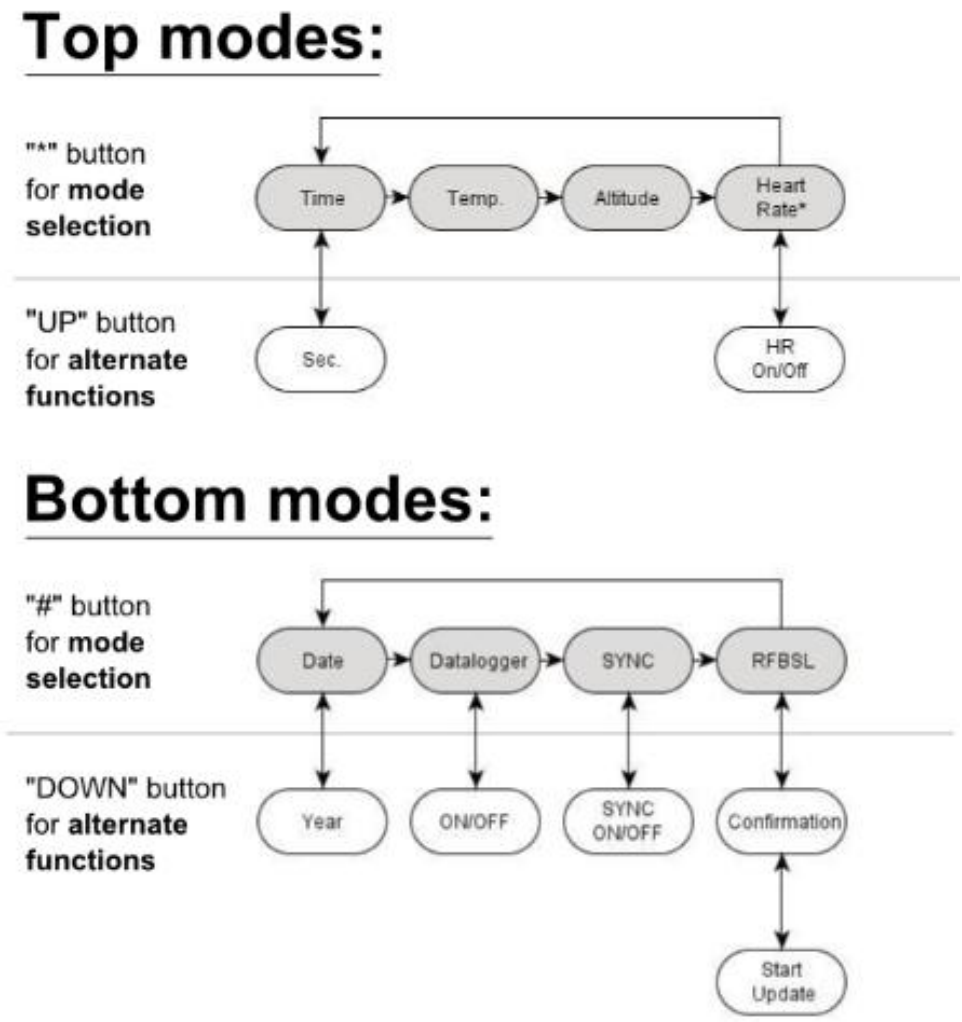
U ACC i PPT načinima rada sat odašilje podatke pristupnoj točki, a pristupna točka tumači podatke ovisno o načinu rada. Kod ACC načina rada sat kontinuirano pristupnoj točki odašilje vrijednosti akceleracije po sve tri osi, a pristupna točka vraća satu potvrdu da je primila podatke. Pri PPT načinu rada, sat ne odašilje podatke kontinuirano, već se podaci prenose sa sata na pristupnu točku samo u trenucima kad korisnik pritisne neku od tipki na satu. Pristupna točka ponovno satu vraća potvrdu da je primila podatke.

U načinu rada SYNC sat šalje pristupnoj točki zahtjev za prijenosom podataka, a pristupna točka odašilje podatke satu. Način rada RFBSL predstavlja bežični update softvera sata. Da bi se koristeći *eZ430-Chronos* razvojni sustav postigao sustav za mjerenje kutne akceleracije i brzine, kao polazište koristit će se postojeći *Data Logger*[6] program umjesto *Sports Watch* programa. *Data Logger* program *Chronos* sata ima minimiziranu funkcionalnost u odnosu na način rada *Sports Watch* s ciljem da se oslobodi što veća količina memorije za pohranu podataka u datalog. Ovaj program zadržava osnovne funkcionalnosti sata kao što su vrijeme i datum, mjerač temperature i visine, ali koristi samo dva načina rada za prijenos podataka. To su SYNC i RFBSL. Za potrebe ovog završnog rada, modificiran će biti SYNC način rada.

U ovom je programu oslobođeno 8 kB flash memorije za pohranjivanje podataka u datalog, što može biti dovoljno za pohranu podataka sa senzora i kroz nekoliko

dana, ovisno o tome kolika je frekvencija uzorkovanja signala sa senzora i veličini podataka koje senzor šalje satu.

Zaslon sata sastoji se od dvije linije. Za kretanje po opcijama gornje linije koristi se tipka označena s „*“, dok se za kretanje po opcijama donje linije koristi tipka „#“. Svaki od načina rada ima i alternativnu funkciju koja se pokreće s tipkom „UP“ za gornju liniju ili tipkom „DOWN“ za donju liniju. Na primjer, za način rada s prikazom vremena alternativna funkcija je prikaz sekundi dok je za način rada SYNC alternativna funkcija početak zahtjeva za prijenosom podataka. Slika 7. prikazuje načine rada Data Logger programa.



Slika 7. Načini rada Data Logger programa

5. Data Logger program

Program se sastoji od 59 .h datoteki i 46 .c datoteki razmještenih po direktorijima ovisno o funkciji u programu. Tako dio datoteki služi kao driver i definira osnovne funkcionalnosti sklopovlja, dio definira protokole za radiofrekvencijski prijenos podataka, *BlueRobin* i *SimpliciTI*, a dio je vezan za logički dio programa.

5.1. Podatkovne strukture

Glavni program, ali i brojne druge funkcije ovog sustava često se pozivaju na prethodno definirane podatkovne strukture i komuniciraju međusobno koristeći strukture kao globalne varijable. U datoteci project.h definirane su tri strukture: *sys*, *request* i *message*.

Struktura *sys* predstavlja sistemske zastavice sustava i nosi tehničke informacije o sustavu. Tu je pohranjena informacija o tome je li korisnik zaključao tipke, da li je baterija prazna, koristi li sustav metričke ili anglosaksonske jedinice, treba li pričekati neko vrijeme prije izvođenja sljedeće naredbe, da li zujalica radi i je li postavljeno vremensko ograničenje u kojem korisnik mora pritisnuti neku od tipki.

Struktura *request* sadrži informaciju o zahtjevima koje procesor ima prema vanjskom sklopovlju. Tu se nalaze zastavice za izvršavanje mjerenja akceleracije, tlaka ili napona baterije te zastavica kojom procesor daje do znanja da želi dodati podatke u datalog.

Struktura *message* sadrži zastavice vezane uz LCD zaslon. Postavljanjem ovih zastavica može se izbrisati prethodnu poruku s ekrana, pripremiti zaslon za ispis sljedeće poruke i prikazati poruku na zaslonu, a koristeći preostalih šest zastavica možemo ispisati šest sistemskih poruka. To su poruke s informacijom da su tipke zaključane ili otključane, poruka da je baterija prazna, poruke „on“ i „off“ te poruka koja izvještava korisnika da je memorija puna.

Za primjer definicije strukturi daje se struktura message. Preostale dvije strukture su definirane na identičan način. Pritom je u16 cjelobrojna poluriječ.

```
typedef union
{
  struct
  {
    u16          prepare                 : 1;
    u16          show                    : 1;
    u16          erase                   : 1;
    u16          type_locked             : 1;
    u16          type_unlocked           : 1;
    u16          type_lobatt             : 1;
    u16          type_on                 : 1;
    u16          type_off                : 1;
    u16          type_nomem              : 1;
  } flag;
  u16 all_flags;
} s_message_flags;
extern volatile s_message_flags message;
```

U datoteci ports.h nalazi se definicija zastavica svih tipki i načina pritiska tipki koje sustav prepoznaje, u strukturi button. Sustav raspoznaje kratki pritisak svih tipki i dugi pritisak tipki „*“ i „#“. Struktura je formirana identično kao i strukture prethodno definirane u datoteci project.h.

Vrlo važna struktura definirana je u datoteci menu.h, a radi se o strukturi svakog od načina rada u meniju. Članovi menija određeni su pozivom četiri funkcije, ovisno o tome koja je tipka pritisnuta, i pokazivačem.

Prva funkcija u svakoj strukturi je sx funkcija i ona se poziva pritiskom tipke „UP“ za gornji red LCD zaslona ili tipkom „DOWN“ za donji red LCD zaslona. Tako primjerice pritisak tipke „UP“ u načinu rada prikaza vremena poziva funkciju sx_time koja na zaslon umjesto sati i minuta ispisuje sekunde. U načinu rada prikaza datuma pritisak na tipku „DOWN“ poziva funkciju sx_date koja na zaslonu umjesto dana i mjeseca prikazuje godinu.

Druga funkcija je funkcija poziva podmenija. Ona se ne koristi niti u jednoj strukturi, osim u bežičnom protokolu *BlueRobin* koji se neće koristiti u ovom završnom radu. Riječ je o mx funkciji.

Treća funkcija je funkcija prikaza LCD zaslona. Ukoliko je došlo do promjena tijekom izvršenja programa i potrebno je osvježiti prikaz na LCD zaslonu, program će pozvati ovu funkciju.

Posljednja je funkcija za ažuriranje podataka. Ovu funkciju pozivamo dugim pritiskom na tipku „*“ za načine rada gornje linije LCD zaslona ili dugim pritiskom na tipku „#“ za načine rada donje linije LCD zaslona. Tako, na primjer, dugim pritiskom na tipku „*“ u načinu rada prikaza vremena možemo urediti vrijeme uređaja, a dugim pritiskom na tipku „#“ u načinu rada prikaza datuma možemo urediti datum uređaja.

Strukture su povezane u kružnu listu koristeći pokazivač. Svaka struktura u sebi posjeduje pokazivač na sljedećeg člana liste, a posljednji član pokazuje na prvog člana.

```
const struct menu menu_L2_Sync =
{
        FUNCTION(sx_sync),
        FUNCTION(dummy),
        FUNCTION(display_sync),
        FUNCTION(update_time),
        &menu_L2_RFBSL,
};
```

Kao primjer se daje definicija strukture menu_L2_Sync, koja će se koristiti za uspostavu bežične komunikacije. L2 u imenu strukture nam govori da je riječ o strukturi koja je dio donjeg reda u meniju LCD zaslona. Osim poziva četiri funkcije, u strukturi je vidljiv poziv na sljedećeg člana, koji je u ovom slučaju način rada bežičnog obnavljanja softvera *Chronos* sata.

Ovaj završni rad koristit će bežični protokol *SimpliciTI* za bežičnu komunikaciju. Taj protokol definira vlastitu strukturu sRFsmpl koja određuje o kojem se načinu prijenosa podataka radi, ACC, PPT, SYNC ili RFBSL i određuje *timeout* interval nakon kojeg se automatski zaustavlja komunikacija. Varijabla mode može imati vrijednosti OFF, BUTTONS, ACCELERATION i SYNC, ovisno o tome u kojem kakve podatke SimpliciTI protokol prenosi.

```
struct RFsmpl
{
	simpliciti_mode_t		mode;
	u16				timeout;
};
extern struct RFsmpl sRFsmpl;
```

5.2. main() funkcija

Dovođenjem napajanja, odnosno, umetanjem baterije pokreće se main funkcija. Dolazi do inicijalizacije aplikacije. Inicijalizacija pokreće watchdog sklop koji resetira sklop svakih šesnaest sekundi ukoliko se dogodi neki problem u izvršavanju programa. Zatim namješta napajanje sklopovlja na 3 V, podešava frekvenciju procesorske jezgre na 12 MHz i određuje koji su priključci ulazni, a koji izlazni. Inicijalizacija zatim izvršava reset radio jezgre i postavlja radio jezgru u način rada smanjene potrošnje. Zatim se izvode inicijalizacije ostalih vanjskih jedinica, a to su akcelerometar, LCD zaslon, tipke, timer i senzor tlaka.

Slijede inicijalizacije globalnih varijabli. Prva inicijalizacija je postavljanje početnog načina rada. Nakon paljenja uređaja, na gornjoj liniji LCD zaslona bit će ispisano vrijeme, a na donjoj liniji LCD zaslona bit će postavljen datalog. To se postiže postavljanjem pokazivača definiranih u strukturama Time i Datalog u varijable definirane u glavnom programu. Nakon toga sustav postavlja stanje svih zastavica u nisku, neaktivnu razinu te obnavlja prikaz na LCD zaslonu. To se čini pristupanjem varijabli all_flags u svim prethodno definiranim strukturama.

Uobičajeno je da sat radi s metričkim jedinicama, ali je moguće promijeniti jedinice u anglosaksonske promjenom zastavice use_metric_units definirane u strukturi sys.

Slijedi baždarenje vrijednosti koje senzori vraćaju, ovisno o eksperimentalnim podacima zapisanim u memoriju ili podacima koje je korisnik sam zapisao u memoriju. Tako mjerač tlaka može određivati nadmorsku visinu uz određenu pogrešku koja se može ispraviti upisom vrijednosti u memoriju. Ukoliko korisnik ne definira vrijednosti za baždarenje sustav, samostalno pokušava otkloniti tu pogrešku eksperimentalnim metodama.

U sklopu inicijalizacije globalnih varijabli slijedi reset svih funkcija koje će uređaj obavljati. Prvo se vrši reset vremena i datuma. Standardno se vrijeme postavlja u 4:30, a datum u 1. 1. 2008. Zatim dolazi reset zujalice, mjerača nadmorske visine, akcelerometra, *BlueRobin* bežičnog protokola, *SimpliciTI* protokola, mjerača napona baterije i dataloga.

Nakon što su dovršene sve inicijalizacije program ulazi u glavnu kontrolnu petlju koja se izvodi sve do prestanka dovođenja napajanja. Program je u načinu rada smanjene potrošnje energije sve dok se ne dogodi neki događaj koji zahtjeva buđenje, a to je obično pritisak neke tipke. Ukoliko procesor dobije nekakav zahtjev, on ga obrađuje te, a ako se promijeni neka od zastavica LCD zaslona, program osvježava prikaz na LCD zaslonu.

```
while(1)
      {
      idle_loop();
      if (button.all_flags || sys.all_flags) wakeup_event();
      if (request.all_flags) process_requests();
      if (display.all_flags) display_update();
      }
```

Funkcija idle_loop omogućava prekide, poslužuje watchdog sklop i ulazi u način rada smanjene potrošnje energije. Izlazak iz te petlje događa se kad neka od vanjskih jedinica pošalje zahtjev za prekidom čime program ispituje sljedeće tri linije koda.

Ukoliko je došlo do promjene neke od sistemskih zastavica ili je pritisnuta neka od tipki, poziva se funkcija wakeup_event.

Funkcija prvo ispituje je li ikoja od tipki pritisnuta i jesu li tipke zaključane te, u slučaju da jesu, ispisuje na zaslonu poruku „Loct“ kojom se korisniku daje do znanja da je tipkovnica zaključana. Ukoliko je tipkovnica otključana i pritisnuta je neka od tipki, funkcija nizom uvjetnih ispitivanja određuje koja je tipka pritisnuta. Ako je korisnik dugo pritisnuo tipku „*“ poziva se druga funkcija iz strukture koja je bila aktivna u trenutku pritiska tipke. Kratkim pritiscima tipki „*“ i „#“ ova funkcija definira kretanje kroz kružnu listu strukturi menija. Pritom se svaki puta osvježava zaslon. Ukoliko je riječ o kratkim pritiscima na tipke „UP“ ili „DOWN“ funkcija

wakeup_event prepoznaje da je korisnik želio pozvati sx funkciju neke strukture te se ona poziva.

Jedina sistemska zastavica koja se uzima u obzir pri ispitivanju u sklopu wakeup_event funkcije je zastavica idle_timeout, odnosno, zastavica koja govori je li isteklo vrijeme u kojem je korisnik trebao pritisnuti neku tipku na satu.

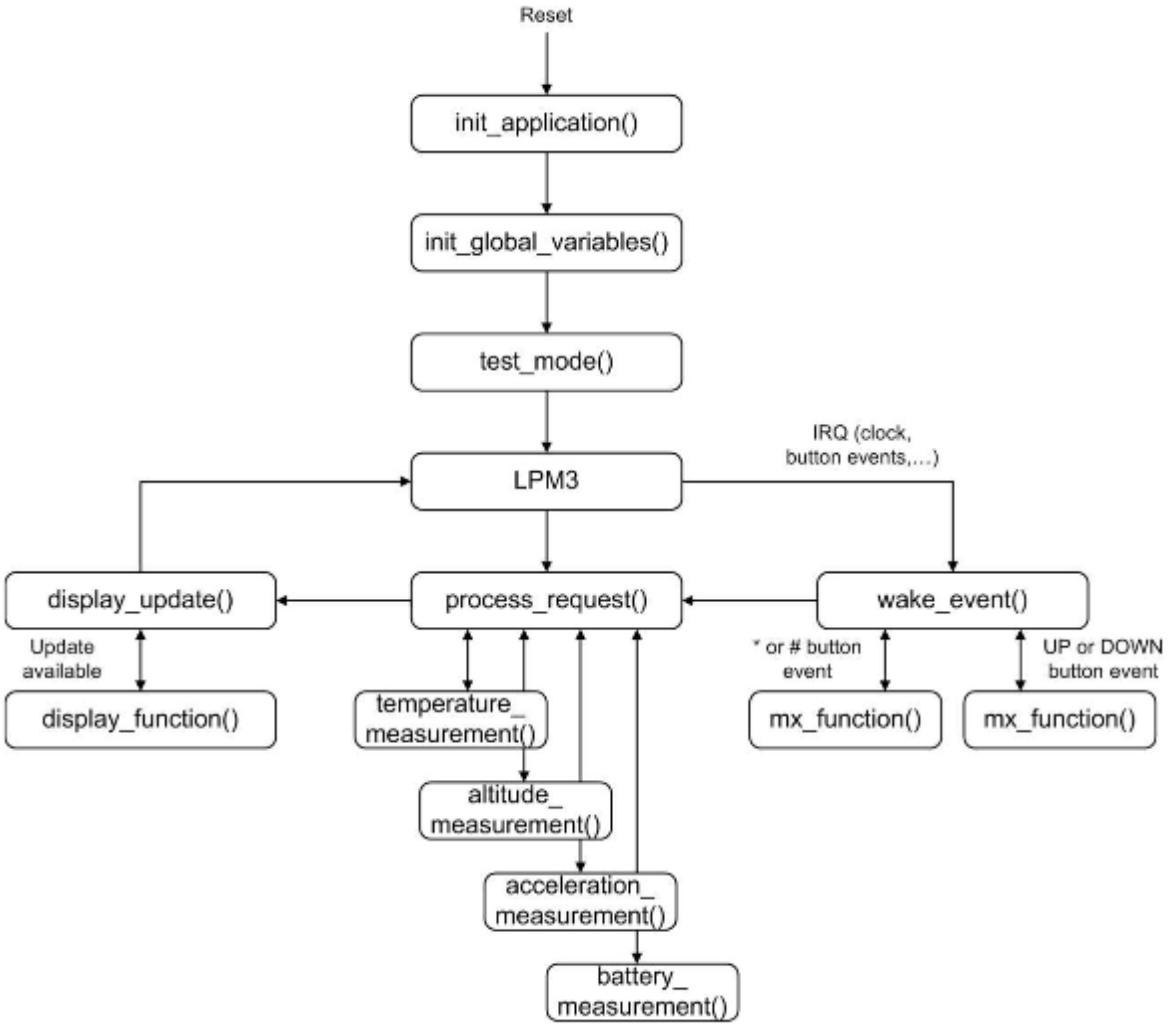
Nakon što su obrađeni svi prekidi koje generiraju tipke, izlazi se iz funkcije wakeup_event te glavni program ispituje da li je došlo do nekog zahtjeva za mjerenjem s nekog od senzora. Primjerice, ukoliko korisnik u načinu rada prikaza temperature pozove funkciju display_temperature postavit će se zastavica s tim zahtjevom u varijabli *request* i main program to prepoznaje i pokušava obraditi taj zahtjev funkcijom process_requests. Ta funkcija obrađuje samo tri različita zahtjeva. Ovisno o tome koji je zahtjev aktivan vrši se mjerenje nadmorske visine preko senzora tlaka, pohranjuju se podaci u datalog ili se radi mjerenje napona baterije.

Povratkom iz funkcije process_requests u main program provjerava se da li je neka od prethodnih funkcija zatražila da se obnovi zaslon ispitujući zastavicu *display*. Ako postoji potreba da se obnovi zaslon taj zahtjev obrađuje funkcija display_update. Ispis na ekran vrši se direktnim kopiranjem stringa.

Time se zatvara kontrolna petlja main programa i ponovno se ulazi u način rada smanjene potrošnje energije sve dok mikrokontroler ne probudi neki sljedeći zahtjev za prekidom.

Ovaj završni rad koncentrirat će se na strukturu SYNC i njenu „DOWN“ funkciju, sx_sync. Ta funkcija služi za odašiljanje zahtjeva za sinkronizacijom i primanje podataka od udaljenog čvora.

Slika 8. prikazuje dijagram toka glavnog programa Chronos sata.



Slika 8. Dijagram toka main() funkcije eZ430-Chronos sata

5.3. sx_sync() funkcija

Funkcija sx_sync može se pronaći u datoteci rfsimpliciti.c, a predstavlja početak protokola bežičnog prijenosa podataka, *SimpliciTI*. Funkcija prvo provjerava je li napon baterije dovoljan za prijenos podataka i je li prethodno uključen neki drugi protokol za prijenos podataka. Uspješnim prolaskom kroz te provjere pokreće se *SimpliciTI* protokol u SYNC načinu rada.

Prije početka prijenosa briše se sadržaj LCD zaslona i zatvara datalog ukoliko je bio uključen. Da bi se pokazalo da je uređaj aktivan iako je zaslon prazan, funkcija postavlja tri ikone na zaslonu da titraju. Zatim se vrše pripreme radio jezgre za

komunikaciju i definira se način prijenosa podataka SYNC u prethodno definiranoj strukturi sRFsmpl.

Pozivom funkcije simpliciti_link sustav pokušava uspostaviti vezu. Ako se veza uspostavi, započinje prijenos podataka glavnom SYNC rutinom, simpliciti_main_sync. Kada je prijenos završen, postavlja se stanje SIMPLICITI_OFF u strukturi sRFsmpl, gasi se radio jezgra, gase titrajuće ikone s ekrana i na ekran se ponovno ispisuje ono što je na njemu do tad pisalo.

5.3.1. simpliciti_link()

Ova je funkcija definirana u datoteci main_ED_BM.c i vezana je za „End Device“. Cilj joj je inicijalizirati sav hardver potreban za vezu i uspostaviti vezu. Vraća jedinicu ako je veza uspješno uspostavljena ili nulu ukoliko veza nije uspostavljena.

Na početku se briše prethodna mrežna adresa i na njeno mjesto upisuje fiksna vrijednost određena u main funkciji. Sve ulazno-izlazne radnje unutar ove funkcije obavljaju se pomoćnom funkcijom SMPL_Ioctl koja nad nekim određenim objektom obavlja neku određenu radnju.

```
SMPL_Ioctl(IOCTL_OBJ_ADDR, IOCTL_ACT_SET, &lAddr);
```

U ovom slučaju se nad objektom adrese obavlja radnja postavljanja.

Tada počinje postupak povezivanja pa se postavlja zastavica simpliciti_flag u stanje SIMPLICITI_STATUS_LINKING. Nakon uspješne inicijalizacije *SimpliciTI* stoga uključuje se radio jezgra i namješta njezina snaga na 3.3 dBm, odnosno 2.14 mW. Koristeći stog, uspostavlja se veza i njezin identitet se sprema u varijablu sLinkID1. U tom je trenutku veza uspostavljena pa se mijenja stanje zastavice simpliciti_flag u SIMPLICITI_STATUS_LINKED.

5.3.2. simpliciti_main_sync()

Nakon što je veza uspostavljena, može započeti prijenos podataka. Funkcija šalje „ready-to-receive“ pakete u stalnim intervalima, zatim osluškuje kratko vrijeme za odgovor s druge strane veze i dekodira primljeni odgovor.

Kontrolna petlja ove funkcije čeka 500 milisekundi između svakog slanja „ready-to-receive“ paketa i zatim, koristeći prethodno definiranu funkciju SMPL_Ioctl, budi radio. Da bi se stimulirao odgovor prvo se šalju dva bajta informacije i čeka odgovor pristupne točke.

```
Timer0_A4_Delay(CONV_MS_TO_TICKS(500));
SMPL_Ioctl( IOCTL_OBJ_RADIO, IOCTL_ACT_RADIO_AWAKE, 0);

ed_data[0] = SYNC_ED_TYPE_R2R;
ed_data[1] = 0xCB;
SMPL_SendOpt(sLinkID1, ed_data, 2, SMPL_TXOPTION_NONE);

SMPL_Ioctl( IOCTL_OBJ_RADIO, IOCTL_ACT_RADIO_RXON, 0);
```

Program zatim čeka sve dok se ne počnu primati podaci od pristupne točke. Kada je prijenos izvršen uspješno, podaci se počinju dekodirati, ovisno o tome što je pristupna točka poslala. Za to se koristi pomoćna funkcija simpliciti_sync_decode_ap_cmd_callback.

Ukoliko je pristupna točka zatražila od sata da joj pošalje podatke preko uspostavljene veze, to se obavlja u sljedećem koraku. Nakon toga se radio jezgra vraća u stanje rada SLEEP, poslužuje se watchdog sklop i zaustavlja se prijenos podataka. Ponovni prijenos započinje tek nakon što prođe definiranih pola sekunde.

5.3.3. simpliciti_sync_decode_ap_cmd_callback()

U SYNC načinu rada svaki paket podataka koji se prenese radiofrekvencijskom vezom bit će veličine 19B i bit će ga potrebno dekodirati da bi se saznalo o kakvim se točno podacima radi. Pristupna točka će u prvi bajt unijeti tip podatka. Recimo,

ukoliko se u prvom bajtu nalazi vrijednost SYNC_AP_CMD_SET_WATCH, to znači da je u sljedeća tri bajta pohranjena informacija o vremenu. U sljedeća je četiri bajta pohranjena informacija o datumu. U preostalim bajtovima se nalaze informacije o nadmorskoj visini i temperaturi. Ti su podaci došli do pristupne točke preko aplikacije koja se izvodi na osobnom računalu.

6. Program pristupne točke

Nakon spajanja pristupne točke na napajanje pokreće se main program pristupne točke. Odmah nakon uključenja napajanja program definira smjer podataka za svaki od ulazno-izlaznih priključaka. Da bi mikrokontroler pravilno funkcionirao, moraju se podesiti postavke oscilatora. Nakon što je oscilator stabilan, mikrokontroler resetira varijablu simpliciti_data u kojoj se nalaze podaci iz prethodnog prijenosa podataka. Zatim mikrokontroler definira prioritete prekida vanjskih jedinica. Najveći prioritet ima USB veza, zatim redom, prvi brojač, radiofrekvencijska veza i četvrti brojač, a sve ostale vanjske jedinice imaju najniži prioritet. Da bi se testirala bežična komunikacija izvodi se jednostavan test i pokreće kontinuirano odašiljanje.

```
test_on=1;
rftest_radio_init();
start_continuous_tx();
while(1);
```

Izvršavanje ovog koda završava tek u trenutku kad je test uspješno izvršen. Glavni program potom inicijalizira USB vezu i omogućava prekide svih vrsta. Program koristi četvrti brojač da bi svakih 1.7 ms poslužio USB vezu šaljući zahtjev za prekidom. Nakon toga je završena inicijalizacija svog sklopovlja te program ulazi u glavnu kontrolnu petlju.

Obzirom da se bežični prijenos podataka može vršiti koristeći različite protokole, kontrolna petlja ispituje koji se protokol koristi u svakom prolazu kroz petlju. Prvo se ispituje *BlueRobin* protokol. Za *BlueRobin* protokol nužno je da je postavljena zastavica bluerobin_start_now i da nisu prethodno uključeni protokol *SimpliciTI* ili protokol za bežično obnavljanje programa. Ukoliko ne treba pokretati prijenos *BlueRobin* protokolom program ispituje je li aktivna zastavica simpliciti_start_now da bi se pokrenuo protokol *SimpliciTI* i na kraju se ispituje je li potrebna bežična obnova programa. Također, moguće je da glavna petlja izvršava provjeru bežične

veze. Ukoliko je postavljena zastavica test_on vršit će se provjera bežičnog odašiljanja podataka.

Obzirom da će ovaj završni rad koristiti protokol *SimpliciTI*, obratit će se veća pozornost ovom dijelu glavne kontrolne petlje.

Nakon što je kontrolna petlja ustvrdila da se želi uspostaviti prijenos *SimpliciTI* protokolom, vrši se podešavanje hardvera i postavlja zastavica system_status u stanje HW_SIMPLICITI_LINKED čime se daje do znanja da je hardver uspješno podešen. Pokreće se glavni program *SimpliciTI* protokola pozivom funkcije simpliciti_main. Kada je prijenos gotov i završi se izvršavanje te funkcije, system_status zastavica mijenja stanje u HW_SIMPLICITI_STOPPED nakon čega program izlazi iz dijela kontrolne petlje vezan za protokol *SimpliciTI* i nastavlja ispitivati je li potreban neki drugi prijenos podataka.

6.1. simpliciti_main()

Funkcija simpliciti_main služi da bi ostvarila programsku vezu i koristeći tu vezu omogućila bežično primanje ili odašiljanje podataka. Funkcija započinje proces uspostavljanja programske veze postavljanjem zastavice simpliciti_flag u stanje SIMPLICITI_STATUS_LINKING. Inicijalizira se *SimpliciTI* stog na isti način kao i u programu ručnog sata, podešava se snaga radio jezgre koristeći funkciju SMPL_Ioctl koja funkcionira na isti način kao i u programu ručnog sata. Zatim program ulazi u kontrolnu petlju.

Kontrolna petlja osluškuje vezu sve dok se ona ne uspostavi. Kada se veza uspostavi, postavlja se zastavica simpliciti_flag u stanje SIMPLICITI_STATUS_LINKED. U tome je trenutku pristupna točka spremna primati podatke sa sata.

Protokol SimpliciTI može raditi u četiri različita način rada, a to su ACC, PPT, SYNC I RFBSL. Paketi podataka u ACC i PPT načinu rada duljine su 4B, dok su u načinu rada SYNC duljine 2B ili 19B.

```
if (SMPL_SUCCESS == SMPL_Receive(linkID0, ed_data, &len))
```

Primljeni podaci pohranjuju se u polje ed_data čije su maksimalne dimenzije 19B, a prima se i podatak o broju primljenih podataka kroz varijablu len. Potom se podaci sortiraju prema broju primljenih bajtova. Paketi podataka duljine 4B obrađuju se ovisno o tome je li riječ o ACC ili PPT podacima. Paketi duljine 2B su *ready-to-receive* paketi, a paketi duljine 19B sadrže informacije.

```
if (getFlag(simpliciti_flag, SIMPLICITI_TRIGGER_SEND_CMD))
                          {
                            clearFlag(simpliciti_flag, SIMPLICITI_TRIGGER_SEND_CMD);
                            len = BM_SYNC_DATA_LENGTH;
                          }
else                      {
                            simpliciti_data[0] = SYNC_AP_CMD_NOP;
                            simpliciti_data[1] = 0x55;
                            len = 2;
                          }
                         SMPL_Send(linkID0, simpliciti_data, len);
                          break;
```

Glavni program ručnog sata u SYNC načinu rada uvijek prvo šalje jedan ready-to-receive paket podataka da signalizira pristupnoj točki da je spreman primiti podatke. Pristupna točka, u trenutku kad primi ready-to-receive paket, traži podatke koje će poslati ručnom satu. Podaci koje pristupna točka odašilje moraju se nalazi u polju podataka simpliciti_data, a može ih postaviti bilo koja vanjska jedinica. Pritom ta vanjska jedinica mora postaviti zastavicu simpliciti_flag u stanje SIMPLICITI_TRIGGER_SEND_CMD.

U originalnom programu vanjska jedinica koja šalje podatke je računalo povezano s pristupnom točkom USB vezom. USB veza postavlja podatke u polje simpliciti_data i postavlja zastavicu simpliciti_flag u stanje SIMPLICITI_TRIGGER_SEND_CMD.

Ako je USB veza postavila podatke u polje simpliciti_data, izvršit će se uvjet if naredbe i traženi podaci će se pomoću funkcije SMPL_Send poslati ručnom satu, koji će ih obraditi na odgovarajući način.

Podaci se iz računala USB vezom prenose koristeći program *Chronos Data Logger* kroz čije se grafičko sučelje mogu definirati podaci za prijenos. Program je izrađen koristeći TCL/Tk za Windows operacijski sustav. Riječ je o skripti koja

poslužuje USB vezu između pristupne točke i računala i omogućuje prijenos podatka između njih.

7. Izvođenje programa sa strane korisnika

Sa stanovišta korisnika uređaj treba kontinuirano raditi i obavljati sve predviđene funkcije. Nakon dolaska napajanja, odnosno, umetanjem baterije, ručnom je satu potrebno dvije sekunde da izvrši sve inicijalizacije i bude spreman za rad. Korisnik u tom trenutku može pritiskom tipke mijenjati načine rada i pokretati određene sporedne funkcije uređaja, kao što su namještanje vremena i datuma, pokretanje štoperice, očitavanje akceleracije s ugrađenog akcelerometra sata ili pokretanje nekog od načina bežičnog prijenosa podataka.

Svaki od načina prijenosa podataka započinje s otprilike sekundu kašnjenja, koliko je potrebno za inicijalizaciju svog hardvera i za definiranje svih parametara prijenosa podataka.

Svi ostali načini rada sata se odvijaju gotovo trenutačno i korisnik ne može primijetiti kašnjenje u izvođenju.

Da bi se izbjegli slučajni pritisci na tipke uređaja, korisnik može zaključati tipkovnicu. To čini držeći pritisnutima tipke „#“ i „DOWN“ tri sekunde.

Kao pomoć u prezentiranju korisniku može poslužiti PPT bežični prijenos podataka koji, kada je aktivan, omogućava korisniku da pritiskom na određenu tipku mijenja aktivni slajd prezentacije u prethodni ili sljedeći. Na sličan način ta opcija omogućava korisniku kretanje kroz zvučne zapise u Windows Media Playeru i iTunes aplikacijama.

8. Zaključak

eZ430-Chronos komunikacijski sustav obuhvaća dva različita komunikacijska kanala uspostavljena između tri procesorske jezgre. Prvi komunikacijski kanal uspostavljen je između procesorske jezgre *MSP430* ručnog sata *Chronos* i mikrokontrolera arhitekture 8051 unutar pristupne točke. Radi se o radiofrekvencijskoj komunikaciji. Drugi je komunikacijski kanal uspostavljen između pristupne točke i osobnog računala, a uspostavljen je upotrebom USB veze.

Sljedeći korak u razvoju sustava za bežično mjerenje kutne akceleracije i brzine bio bi pokušaj uspostave alternativne komunikacije između računala i pristupne točke preko vlastite aplikacije.

Konačni sustav posve bi uklonio posredovanje osobnog računala u komunikaciji. Izravnim spajanjem senzora na priključke pristupne točke omogućilo bi se odašiljanje akcelerometra ručnom satu radiofrekvencijskom vezom.

U tom slučaju se pristupna točka više ne bi mogla napajati koristeći USB vezu pa bi se moralo ostvariti alternativno baterijsko napajanje.

9. Literatura

[1] Texas Instruments, *eZ430-ChronosTM Development Tool – User's Guide*, 10. 6. 2013., http://www.ti.com/lit/ug/slau292d/slau292d.pdf.

[2] Texas Instruments Wiki, *eZ430-Chronos – Texas Instruments*, 10. 6. 2013., http://www.processors.wiki.ti.com/index.php/EZ430-Chronos.

[3] Texas Instruments, Sub-1 GHz RF System-on-Chip with Integrated USB, 10. 6. 2013., http://www.ti.com/corp/docs/landing/cc1111/.

[4] Texas Instruments, *Debugger and Programmer for RF System-on-Chips*, 10. 6. 2013., http://www.ti.com/tool/cc-debugger.

[5] Texas Instruments, *CC Debugger User's Guide (Rev. E)*, 10. 6. 2013., http://www.ti.com/lit/ug/swru197e/swru197e.pdf.

[6] Texas Instruments, *Chronos Data logger binary*, 10. 6. 2013., http://www.ti.com/lit/zip/slac341.

10. Sažetak

eZ430-Chronos je razvojno okružje koje omogućava bežičnu komunikaciju između dvije krajnje točke. Komunikacija se odvija koristeći dva prethodno definirana protokola, *SimpliciTI* i *BlueRobin*, u različitim načinima rada. Riječ je o sustavu sa smanjenom potrošnjom energije.

Chronos sat radi smanjenom potrošnjom sve dok ga ne probudi pritisak neke tipke, nakon čega obavlja odgovarajuću radnju. Komunikacija započinje uspostavom bežične veze između sata i pristupne točke.

Pristupna točka radi smanjenom potrošnjom energije sve dok ne zaprimi zahtjev za komunikacijom od *Chronos* sata. Ovisno o tipu zaprimljenog zahtjeva, pristupna točka odašilje različite podatke.

Komunikacija je optimizirana tako da se omogući što dulje trajanje baterije.

11. Summary

eZ430-Chronos is a development environment that allows wireless communication between two endpoints. Communication occurs using two predefined protocols, *Simpliciti* and *BlueRobin*, in various modes. This is a system with reduced power consumpion.

Chronos watch works in reduced power mode until it wakes up after a button has been pressed and then performs the appropriate action. Communication begins by establishing a wireless connection between the watch and the access point.

Access point works in reduced energy consumption until receipt of the request for communication from Chronos watch. Depending on the type of request of the application, the access point transmits different data.

Communication is optimized so as to allow more battery life.