

IOT MERNO-INFORMACIONI SISTEM ZASNOVAN NA FREERTOS OPERATIVNOM SISTEMU**IOT MEASUREMENT AND INFORMATION SYSTEM BASED ON FREERTOS OPERATING SYSTEM**Stefan Tešanović, *Fakultet tehničkih nauka, Novi Sad***Oblast- ELEKTROTEHNIKA I RAČUNARSTVO**

Kratak sadržaj – *Ovaj rad predstavlja merno-informacioni sistem koji ima ulogu male meteorološke stanice.*

Cljučne reči: *IoT, FreeRTOS, merenje, akvizicija*

Abstract – *This paper presents a measurement and information system that acts as a small meteorological station.*

Keywords: *IoT, FreeRTOS, measurement, acquisition*

1. UVOD

Uticaj IoT (*Internet of things*) se može videti u gotovo svim sferama današnjeg vremena, bilo da je to industrija, medicina, poljoprivreda pa sve do svakodnevnog upotrebe u domaćinstvu. Njegova uloga je da se uređaji koji u normalnom slučaju ne komuniciraju spoje i samostalno izvršavaju radnje, uzimajući u obzir podatke koje dobijaju od drugih uređaja.

Razlozi zbog kojih se sve više koristi ovaj koncept su: sve se dešava u realnom vremenu, mnogo je veća bezbednost korisnika, automatski je i uz manje napora. Ovo su samo neke od prednosti IoT u odnosu na pojedinačne uređaje, ali pored prednosti postoje i mane.

Neke od značajnijih mana su: memorijski proctor (sa povećanjem broja uređaja potrebno je i više mesta za skladištenje svih podataka), sigurnost podataka (potencijalna meta hakera), greške ili bagovi (*bugs*) koji se mogu javiti pri komunikaciji i kompatibilnost uređaja u sistemu.

U radu je opisan projekat koji ima ulogu male meteorološke stanice. Sastoji se iz dva dela, hardverskog (četiri senzorska modula, EasyPIC Fusion razvojna ploča sa mikrokontrolerom i računara) i softverskog dela (server, baza podataka i veb stranica).

Senzorski moduli postavljeni su na ploču koja na sebi ima mikrokontroler, a razvojna ploča je povezana sa računarem putem serijske komunikacije. Računar koji ima ulogu servera prima podatke, parsira ih i upisuje u bazu podataka. Podaci se uzimaju iz baze podataka i prikazuju na veb stranici u vidu tabele i grafika.

2. HARDVER

Ključna komponenta hardverskog dela sistema jeste razvojna ploča EasyPIC Fusion v7, razvijena od strane kompanije Mikroelektronika. Ploča podržava mikroprocesore iz porodica dsPIC33, PIC24 i PIC32, a na sebi poseduje mnoge module.



Sl 1. Izgled EasyPIC Fusion v7 razvojne ploče

U projektu su od modula koje ploča poseduje korišćeni mikroBUSTTM soketi i UART konektor, mikroprocesor je iz porodice PIC32, odnosno PIC32MZ2048ECH144.

2.1. Senzorski moduli

Projekat ima četiri senzorska modula koji šalju podatke mikrokontroleru. Podaci se mogu primati istovremeno sa više senzora, a korisnik može da odluči na koliko će se sekundi podaci slati mikrokontroleru, koji ih dalje obrađuje. Na slici 2. su prikazani korišćeni senzorski moduli: Illuminance click, CO click, DHT22 click i Barometer click.



Sl 2. Senzorski moduli

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Platon Sovilj.

Illuminance click u sebi poseduje TSL2561 konvertor svetlosti u digitalnu vrednost sa senzorom dizajniranim da

oponaša ljudsku percepciju svetlosti. Većinski se koristi kod aplikacija koje mere ambijentnu svetlost. Senzor ima dve fotodiode, jedna koja je osetljiva na ceo spektar svetlosti, a druga na infracrveni spektar. Vidljivi spektar se dobija preko formule. Pločica komunicira sa mikrokontrolerom putem I2C komunikacije, a napaja se naponom od 3.3 V [4].

CO click u sebi nosi **MQ-7** senzor za detekciju CO gasa. Opseg detekcije CO se kreće od 20 ppm do 2000 ppm. Komunicira sa mikrokontrolerom preko AN pina i napaja se sa 5 V. Takođe postoji potencijometar za kalibraciju senzora [3].

DHT22 click je pločica za merenje vlažnosti i temperature na kojoj se nalazi istoimeni senzor. Senzor može da detektuje temperature u opsegu od -40 °C do 80 °C sa preciznošću od pola stepena, dok se relativna vlažnost meri od 0 % do 100 % sa preciznošću od 2 %. Senzor može da se napaja sa 3.3 V ili 5 V [1].

Barometer click je senzor visoke preciznosti sebi ima **LPS25HB** integrisano kolo koje daje 24-bitne vrednosti pritiska i temperature. Opseg za merenje pritiska je od 260 hPa do 1260 hPa. Preciznost ovog senzora može biti i do 0.01 hPa ukoliko se nalazi u visoko rezolucionom modu. Napon napajanja senzora je 3.3 V i može da komunicira putem SPI ili I2C komunikacije [2].

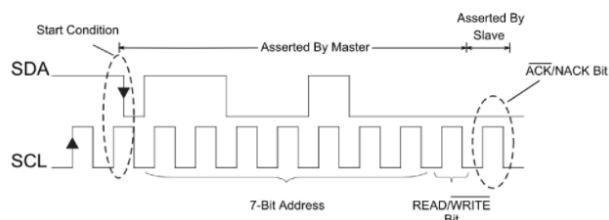
2.2. Komunikacije

Kako bi podaci mogli da se sa senzora šalju mikrokontroleru, pa onda dalje računaru, potrebno je napraviti odgovarajući vid komunikacije između uređaja. To znači da se u zavisnosti od zahteva uređaja i dostupnosti podataka bira odgovarajuća vrsta komunikacije. Projekat u sebi ima nekoliko vidova komunikacije i mogućnost zamene za neki drugi vid ukoliko korisnik to želi.

1-wire (One-wire) vrsta komunikacije koristi jednu liniju za komuniciranje. Jedan uređaj može biti master, a slave mogu biti jedan ili više uređaja. Komunikacije se odvija samo u jednom smeru istovremeno (half duplex), a brzina prenosa standardnih 15 kbps ili 111 kbps u specijalnim slučajevima. Što se tiče komunikacije sa mikrokontrolerom, 1-wire ima četiri operacije: Reset, Write 0 (Upis 0), Write 1 (Upis 1) i Read (Čitanje).

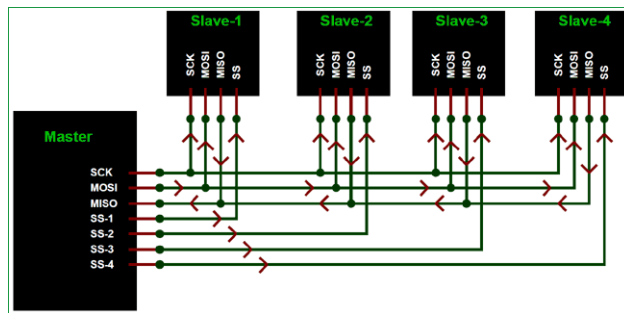
I2C (Inter-Integrated Circuit) je komunikacioni protokol dizajniran od strane kompanije Philips semiconductors 1980-tih godina kako bi olakšao komunikaciju između centralnog procesora i čipova na periferiji. Služi za komunikaciju na maloj razdaljini i koristi dve linije za komunikaciju u oba smera, SDA (Serial Data line) i SCL (Serial clock line). Na linije mogu biti zakačeni više master i više slave uređaja, komunikacija je sinhrona i zbog toga što koristi taktnu liniju može da ostvari velike brzine prenosa. Na slici 3. prikazan je princip rada I2C protokola.

SPI (Serial Peripheral Interface) je komunikacioni protokol namenjen za komuniciranje uređaja na maloj razdaljini. Koristi četiri linije za komunikaciju, što ga čini full-duplex komunikacijom, omogućavajući komunikaciju između master i slave uređaja istovremeno. Slave uređaji ne mogu direktno da komuniciraju sa drugim slave uređajima, odnosno komunikacija mora biti između master i slave uređaja.



SL 3. Princip rada I2C komunikacionog protokola

Linije potrebne za komunikaciju su SCK (Serial Clock), MOSI (Master Out Slave In), MISO (Master In Slave Out) i SS (Slave Select). Slika 4. prikazuje primer SPI komunikacije jednog master i tri slave uređaja.



SL 4. SPI komunikacija između master i slave uređaja

SPI i I2C komunikacije imaju svoje prednosti i mane, zbog toga se koristi onaj protokol koji više odgovara aplikaciji ili projektu. I2C je adekvatniji ukoliko postoji mnogo slave uređaja i koristi samo dve linije za komunikaciju, što ga čini konstruktivno jeftinijim rešenjem, dok SPI koristi četiri linije i za razliku od I2C, komunikacija može da budu istovremeno između master i slave i slave i master uređaja.

3. SOFTVER

U ovom odeljku opisan je FreeRTOS, njegove prednosti i neke od njegovih primena u projektu.

FreeRTOS (Free Real Time Operating System) je API koji za ulogu ima pokretanje više aplikacija istovremeno tako što ih deli u thread-ove, što je odlika današnjih operativnih sistema po čemu je dobio ime. Podrazumeva se da procesor može da vrši jednu radnju u datom trenutku, ali ukoliko postoji scheduler koji putem redosleda i prioriteta izvršavanja daje znak procesoru koju radnju treba da izvršava, smanjuje vreme koje je potrebno za izvršavanje više aplikacija. To daje prividan osećaj da se više aplikacija vrši istovremeno, pa se zbog toga kaže da on radi u realnom vremenu.

Kada neki uređaj ili sistem radi u realnom vremenu, tom sistemu se poznaje vremenski interval u kom će u se radnja izvršiti. Sistemi koji rade u realnom vremenu mogu se podeliti u tri grupe:

1. **Soft Real Time** sisteme kod kojih radnja može da izađe iz zadatog vremenskog interval, ali tako da sistem radi funkciju za koju je namenjen.
2. **Hard Real Time** sisteme kod kojih radnja mora da se obavi u datom vremenskom intervalu.

3. **Soft i Hard Real Time** sistemi koji imaju odlike prve i druge grupe u zavisnosti koliko je sistem ili deo sistema siguran. U ovu grupu spadaju i embeded sistemi.

3.1. Task (zadatak) funkcija

Ova funkcija predstavlja osnovnu funkciju za izvršavanje neke radnje, nema povratnu vrednost i najčešće se izvršava u beskonačnoj petlji, sa mogućnošću brisanja zadatka ukoliko više nije potreban. Takođe, jedan zadatak može biti iskorišćen za kreiranje više resursno nezavisnih zadataka koji se odvojeno izvršavaju.

Aplikacije u najčešćem broju slučajeva koriste više zadataka koji rade istovremeno, tzv. multitasking. Problem kod sistema koje koriste jedno jezgro, kakav je u ovom projektu, je to što se samo jedna radnja može izvršavati u datom trenutku.

Kako bi se ovaj problem ublažio uvode se dva stanja za svaki zadatak, stanje kad se on izvršava i stanje kada čeka da bude izvršen. Pored uvođenja stanja dodaje se i FreeRTOS Scheduler, odnosno raspoređivač koji ima ulogu da odabere kada će se koji zadatak izvršavati.

Sledeći problem nastaje ako je neki zadatak bitniji od drugog. Rešenje tog problema je parametar pri kreiranju task funkcije, odnosno parametar zadužen za podešavanje prioriteta nekog zadatka.

Ukoliko je prioritet nekog zadatka veći, taj zadatak može da zabrani izvršavanje zadatka manjeg prioriteta dok se prvi zadatak ne završi ili da prekine izvršavanje zadatka sa manjim prioritetom ako je prvi zadatak bitniji od drugog.

Scheduler će uvek birati zadatak najvišeg prioriteta nakon svakog generisanog prekida (interrupt).

U stanju čekanja zadatak se može nalaziti u jednom od tri stanja, odnosno blocked (blokiranom), suspended (zabranjenom) ili u ready (spremno) stanju.

Blokirano stanje predstavlja stanje u kome se nalazi zadatak dok čeka neki događaj da se desi nakon koga se zadatka izvršava. Događaji se mogu dešavati u jasno definisanom vremenskom intervalu ili nakon što im prethodi drugi događaj.

Ovim stanjem se rešava problem beskonačnog izvršavanja zadatka sa najvišim prioretom, odnosno dok je zadatak sa višim prioritetom u ovom stanju mogu da se izvršavaju zadaci nižeg prioriteta.

Zabranjeno stanje je stanje kada je zadatku onemogućen pristup za Scheduler i samim tim zadatak se ne može izvršavati. Zadatak se prebacuje u ovom stanje pomoću funkcije `vTaskSuspend()`, a iz stanja se izbacuje pozivom `vTaskResume()` ili `vTaskResumeFromISR()` funkcije. Često se izbegava upotreba ovog stanja u aplikacijama.

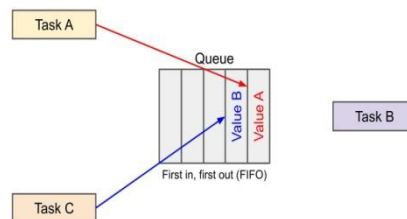
Zadatak je u spremnom stanju ako nije niti u blokiranom niti u zabranjenom stanju i čeka poziv od Schedulera kako bi se izvršio. Kada je zadatak izabran, prebacuje se u stanje izvršavanja.

Ukoliko se zadatak ne nalazi u stanju izvršavanja ili u stanju čekanja onda se sistem nalazi u stanju pripravnosti (Idle state). Stanje pripravnosti je stanje kada procesor nema aktivnih zadataka za izvršavanje, stoga on izvršava tzv. Idle Task.

To je zadatak najmanjeg prioriteta i u taj zadatak korisnik može da pozove funkciju Idle Task Hook koja može da se poziva periodično.

3.2. Queues (Redovi)

Redovi predstavljaju bazicni mehanizam komunikacije i sinhronizacije zadataka. S obzirom da nisu vezani ni za jedan zadatak, imaju mogućnost da dobijaju podatke iz više redova istovremeno, što važi i za slanje podataka u druge redove. Ukoliko ne postoje potpuni podaci za čitanje, zadatak zadužen za čitanje podataka se nalazi u blokiranom stanju, a kada pristignu podaci, automatski se zadatak prebacuje u spremno stanje. Takođe ukoliko je posle isteka vremena, koje postavlja korisnik, red prazan zadatak prelazi u spremno stanje. Na slici 5. Prikazan je primer red u kome dva zadatka upisuju, a jedan zadatak čita podatke.



SL 5. Primer reda

Ako aplikacija ima više zadataka koji čitaju iz reda, samo će jedan zadatak moći da pročita podatak u datom trenutku i to je zadatak sa najvećim prioritetom, a ako više zadataka imaju isti prioritet tada podatke čita zadatak koji je duže čekao u redu.

Kada zadatak koji upisuje podatke pokuša da upiše u red koji je pun, prebacuje se u blokirano stanje dok se ne oslobodi mesto u redu. Čim se mesto u redu oslobodi, zadatak prelazi u spremno stanje. Problem koji se nameće je šta ako postoji aplikacija u kojoj više zadataka upisuje u jedan red, odnosno kako da znamo koji je zadatak poslao koji segment u tom redu. Rešenje problema je kreiranje struktura podataka koje će da sadrži podatke i informacije o zadatku koji je te podatke poslao.

Kada je potrebno da se zaustavi neki zadatak koji je u stanju izvršavanja, koristi se funkcija `vTaskYield()`, koja prebacuje zadatak u blokirano stanje i Scheduler na njegovo mesto postavlja zadatak sa isti ili zadatak višeg prioriteta, ali ako ne postoji takav zadatak, tj. ostali zadaci su manjeg prioriteta, prvi zadatak se ponovo izvršava.

3.3. Semaphores (Semafori)

Semafori se uglavnom koriste događaji koji se dešavaju nakon prekida. Imaju dve uloge: giving (davanje) i taking (uzimanje). Uloga uzimanja je da postavi zadatak u blokirano stanje dok se ne desi određeni događaj, a uloga davanja je da nakon prekida prebaci zadatak iz blokiranog u spremno stanje. Takođe postoji i Handler task koji u zavisnosti od uloge semafora postavlja zadatak u adekvatno stanje. Semafori se dele na binarne i brojačke.

Binarni semafori predstavljaju red koji ima jedan element. Ukoliko je red prazan i zadatak pokuša da čita iz tog reda, zadatak će biti poslan u blokirano stanje. Nakon što se desi prekid i generisana je funkcija davanja, red dobija token i tada je popunjen. Zatim se izvršava Handler task koji zadatka koji treba da čita iz reda šalje u spremno stanje. Kada zadatak pročita iz reda, ponovo se šalje u blokirano stanje. Problem kod ovakvog načina rada apli-

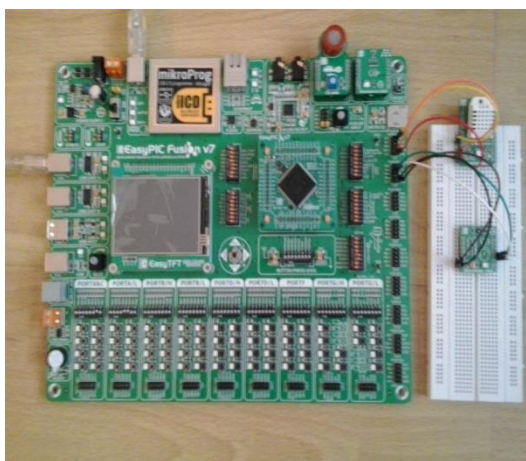
kacije jeste to što se prekid može izvršiti samo jednom tokom izvršavanja zadatka, a ako se desi da više prekida, tada se gube događaji.

Rešenje problema je upotreba brojačkog semafora. Za razliku od binarnog semafora koji se posmatra kao red sa jednim elementom, brojački semafor se posmatra kao red sa više istih. Oni se koriste u dva slučaja: za brojanje događaja i za upravljanje resursima. Za brojanje događaja se koriste tako što svaki prekid generiše funkciju davanja koja dodaje element u red, dok se funkcijom uzimanja, odnosno čitanjem iz reda, smanjuje broj elemenata u redu. Uloga upravljanja resursima je takva da broj elemenata u redu predstavlja broj raspoloživih resursa, kada zadatak uzme element iz reda smanjuje se broj resursa, dok se red ne isprazni, a kada zadatak završi sa resursom vraća ga na kraj reda [5].

5. TESTIRANJE SISTEMA

Za početak je potrebna kalibracija senzorskih modula i njihovo povezivanje na razvojnu ploču EasyPIC Fusion v7. Senzorski moduli su povezani sa pločom putem mikroBUS™ konektora, ali pošto na ploči postoje dva takva konektora, druga dva senzora su povezana direktno preko pinova. Nakon spajanja senzorskih modula, povezuje se razvojna ploča sa računarom putem UART konektora, odnosno USB priključka na računaru.

Kako bi senzorski moduli slali podatke, potrebno je spustiti kod za svaki od senzora na ploču i na kraju je potrebno napisati odgovarajući kod za hardverski deo aplikacije. Taj kod se sastoji od postavljanja zadataka za svaki senzor pojedinačno i zadatka za slanje podataka računaru. Postavljeni su odgovarajući prioriteti na zadatke kako ne bi došlo do gubljenja podataka i postavljeni su tajmeri za svaki od senzora. Ovaj deo koda napisan je u jeziku MikroC za PIC32 kompanija MikroElektronika. Izgled povezane ploče sa sensorima prikazan je na slici 6.



SL 6. Razvojna ploča sa senzorkim modulima

Kada podaci stignu na računar potrebno ih je parsirati i smestiti u bazu podataka. Za ovaj deo projekta korišćen je programski jezik Python. Kreirane su dve aplikacije, jedna koja ima ulogu da primi podatke i parsira ih, dok druga pokreće bazu podataka u upisuje podatke u nju.

Prva aplikacija posmatra da li mikrokontroler šalje podatke i ako je to slučaj razdvaja ih u zavisnosti od kog su senzorskog modula došli i dodaje je im vreme i datum po kome će podaci biti sortirani. Ova aplikacija proverava da li su pristigli podaci svake sekunde i nakon što ih parsira, pravi rečnik u kome upisuje vrednosti i raspoređuje ih na odgovarajući način.

Druga aplikacija koristi MongoDB bazu podataka, ona proverava da li su podaci odgovarajućeg formata i upisuje ih u bazu podataka. Za preuzimanje podataka iz baze i pokretanje veb stranice korišćeni su JavaScript, HTML i CSS. Podaci se uzimaju iz baze podataka, prave se odgovarajući grafici i tabele, određuju se minimalne, maksimalne i srednje vrednosti koje se predstavljaju na veb stranici.

6. ZAKLJUČAK

Opisani projekat uz male modifikacije može da se koristi i u komercijalne svrhe, trenutno je namenjen za kućnu upotrebu, takođe projekat nije ograničen na senzorske module koji su korišćeni u njemu. Dakle, moguće je korišćenje drugih senzorskih modula kao i merenje drugih veličina uz odgovarajuću opremu..

Prednosti ovakvog sistema su njegova jednostavnost i prikaz podataka u realnom vremenu. Sistem je napravljen da radi u lokalnoj mreži, ali postoji mogućnost povezivanja na globalnu internet mrežu. U tom slučaju korisnik može lakše da pristupi podacima dobijenim od senzora, ali tada bi mogla da se pojavi potencijalna pretnja korišćenja podataka od strane trećeg lica koje ne bi trebalo da ima pristup podacima. Ukoliko bi korisnik želeo, sistem je moguće napajati baterijskim izvorom napajanja. Tako bi on dobio na mobilnosti, ali baterije nisu namenjene da traju duži vremenski period, odnosno stalna kupovina baterija bi bila dodatni trošak.

Kao poboljšanje, sistemu se može dodati aplikacija ili deo aplikacije koji prikazuje podatke iz nekog vremenskog perioda, kao grafički interfejs za lakši pristup podacima. Sistemu se takođe može dodati aplikacija koja posmatra da li je veličina u dobrom opsegu i prijavi korisniku da je neka od veličina van opsega ili da primeni odgovarajuće mere ukoliko korisnik to dozvoli.

7. LITERATURA

- [1] <https://www.mikroe.com/dht22-click>
- [2] <https://www.mikroe.com/barometer-click>
- [3] <https://www.mikroe.com/co-click>
- [4] <https://www.mikroe.com/illuminance-click>
- [5] Materijali iz predmeta Merenja u realnom vremenu

Kratka biografija:

Stefan Tešanović rođen je 1996. godine u Novom Sadu. Diplomirao je na fakultetu tehničkih nauka u Novom Sadu na katedri za električna merenja 2019. godine.