Seminarski rad iz predmeta Metode i alati za opis hardvera

Implementacija S-MAC protokola za bežične senzorske mreže na FPGA

Enio Kaljić, mr.el.-dipl.ing.el.
Elektrotehnički fakultet Univerziteta u Sarajevu

22.09.2013

Sadržaj

Uvod

U realizaciji čvorova bežične senzorske mreže (eng. *Wireless Sensor Network*, WSN) postoje dva pristupa: upotreba mikrokontrolera i RF (eng. *Radio Frequency*) periferije, te realizacija cijelog sistema na čipu. U prvom pristupu, na mikrokontroleru se izvršava neki real-time operativni sistem ili namjenski pisani softver, a RF periferija se kontroliše pomoću komunikacijskih protokola koji se također izvršavaju na mikrokontroleru. Navedeni pristup je veoma jeftin i uglavnom zadovoljava zahtjeve aplikacije. U drugom pristupu, na jednom čipu se realizuju sve komponente i funkcionalnosti sistema (procesor, memorija, obrada signala, protokoli, itd.). Rezultat ovog pristupa je aplikacijski definisano integrisano kolo (eng. *Application-Specific Integrated Circuit*, ASIC). Prednost ASIC-a u odnosu na mikrokontrolersko rješenje leži u mogućnosti optimizacije potrošnje električne energije i realizaciji samo onih funkcionalnosti koje su tražene datom aplikacijom. Danas, najpopularniji način realizacije prototipa ASIC-a je upotreba FPGA (eng. *Field-Programmable Gate Array*) tehnologije.

U realizaciji bežičnih senzorskih čvorova (eng. *Wireless Sensor Node*, WSN) na FPGA čipu velik broj autora pribjegava upotrebi mikroprocesorskih arhitektura (Altera Nios, Xilinx PicoBlaze i sl.). U ovom slučaju, komunikacijski protokol (najčešće protokol za kontrolu pristupa mediju) se realizuje u nekom programskom jeziku (n.pr. C/C++) i izvršava se na mikroprocesoru realiziranom na FPGA čipu. Međutim, u ovom radu je postavljena sljedeća hipoteza: protokol za kontrolu pristupa mediju je moguće realizirati kao hardversku strukturu koja će upotrijebiti manje logičkih elemenata FPGA čipa nego što bi bilo potrebno za realizaciju najjednostavnije Altera Nios mikroprocesorske arhitekture. Cilj ovog rada je implementirati protokol za kontrolu pristupa mediju za bežične senzorske mreže na FPGA i provjeriti postavljenu hipotezu.

U prvom dijelu rada je dat pregled najvažnijih faktora pri dizajnu protokola za kontrolu pristupa mediju u WSN-u. U drugom dijelu rada je odabran protokol koji najbolje odgovara na postavljene zahtjeve, a zatim je upotrebom SDL (eng. *Specification and Description Language*) dijagrama urađen dizajn protokola. U trećem dijelu je dat kratki pregled arhitekture sistema, specifično protokola za kontrolu pristupa mediju i protokola fizičkog sloja prema referentnom OSI/ISO modelu. U četvrtom dijelu je odabrana hardverska platforma za implementaciju i testiranje protokola, te je dat pregled najvažnijih karakteristika korištenog FPGA čipa. U konačnici je dat pregled komponenti sistema koje su specificirane upotrebom VHDL jezika.

Dizajn protokola za kontrolu pristupa mediju

Važni faktori pri dizajnu protokola za kontrolu pristupa mediju (eng. *Medium Access Control*, MAC) u bežičnim senzorskim mrežama su:

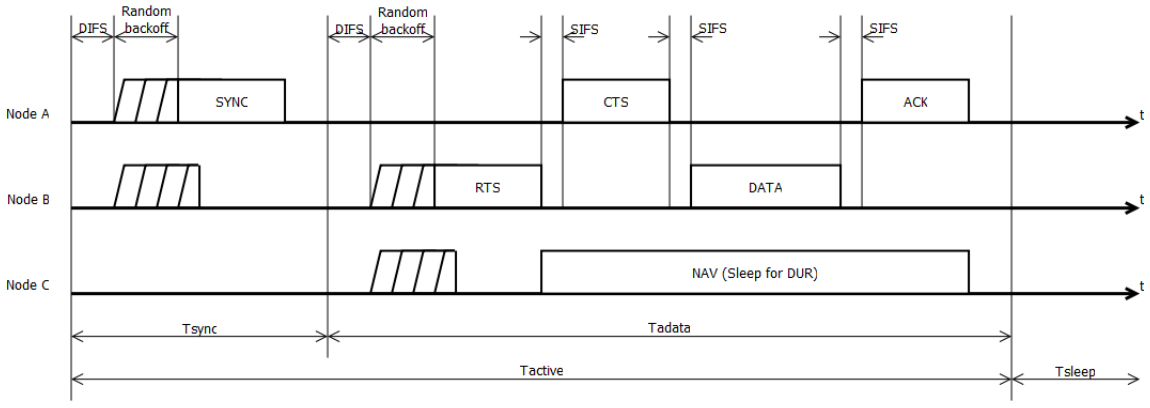
- Ušteda energije - glavni cilj pri dizajnu MAC protokola za bežične senzorske mreže je ušteda energije na bežičnim senzorskim čvorovima čime se povećava životni vijek mreže. Važan faktor je također i QoS u mreži, koji se manifestuje kroz parametre kao što su propusnost, kašnjenje, pravednost, iskorištenje propusnog opsega itd. Da bi se ispunili spomenuti ciljevi potrebno je dizajnirati efikasan protokol na MAC sloju. [1] Procesi koji u značajnoj mjeri utiču na potrošnju energije na ovom sloju jesu [2]:
  - Kolizije - kada se desi kolizija, oštećeni paket se odbacuje te se vrši retransmisija, što rezultira višestrukim prenosima istih podataka i povećanom potrošnjom energije. Iz svega navedenog proizlazi da MAC protokol mora predvidjeti mehanizam za izbjegavanje kolizije.
  - Stalno osluškivanje medija - čvorovi koji stalno osluškuju medij primaju pakete namijenjene drugim mrežnim čvorovima, što povećava potrošnju energije osobito u mrežama sa velikim brojem čvorova.
  - Osluškivanje medija u IDLE stanju - čvorovi u IDLE stanju (besposleni) moraju osluškivati medij, zbog potencijalne transmisije njima namijenjenih podataka. Iako čvor ne prima niti predaje ovim se značajno povećava potrošnja energije.
  - Prenos kontrolnih paketa - prenos, prijem i osluškivanje kontrolnih paketa uzrokuju dodatnu potrošnju energije, stoga je važno da se broj kontrolnih paketa u mreži svede na minimum.
  - Prenos prema nedostupnim čvorovima - ukoliko je paket upućen nedostupnim čvorovima dolazi do nepotrebne potrošnje. Drugim riječima potrebno je da čvorovi u mreži imaju informacije o dostupnosti drugih čvorova.
- Vremenska osjetljivost – drugi važan zahtjev pri dizajnu MAC WSN-a jeste osiguravanje prenosa u realnom vremenu za aplikacije koje to zahtijevaju. [2]
- Skalabilnost i prilagodljivost – MAC protokol bežične senzorske mreže mora biti visoko skalabilan zbog velikog broja senzorskih čvorova i različitih funkcionalnosti koje mreža može podržavati. Također mora biti prilagodljiv dinamičkim promjenama topologije koje nastaju zbog ispada i/ili mobilnosti čvorova. Mreža bi trebala biti i samoorganizirajuća i sposobna za normalnu funkcionisanje bez obzira na nepouzdanost pojedinog čvora. Da bi se mreža učinila pouzdanijom poželjna je decentralizirana arhitektura mreže. [2]
- Kompleksnost – odnosi se na potrošnju energije pri izvršavanju procesorski kompleksnih algoritama i protokola. Jedan od veoma važnih zahtjeva pri dizajnu MAC WSN-a je i jednostavnost. [3]

S-MAC protokol

U postupku dizajna protokola potrebno je pronaći kompromis između gore postavljenih zahtjeva, budući da su neki od njih u suprotnosti sa drugim. Protokoli koji se koriste u okviru MAC sloja WSN mreža se u opštem slučaju klasificiraju u dvije kategorije: protokoli bazirani na natjecanju za medij i TDMA protokoli. Iako se IEEE 802.11 protokoli uglavnom koriste zbog jednostavnosti i otpornosti na problem skrivenih čvorova oni su energetski neefikasni jer ne rješavaju probleme stalnog osluškivanja medija i osluškivanja u Idle stanju. Detalji o nedostacima koji eliminišu pojedine protokole iz gore navedenih kategorija u odnosu na S-MAC protokol su opisani u [4]. Pregled svih postojećih WSN MAC protokola zajedno sa njihovim prednostima i nedostacima također je dat u [6].

Glavna prednost S-MAC protokola je stanje spavanja koje značajno reducira potrošnju energije. Protokol se također lahko adaptira na promjene topologije. Dodatno, nema potrebe za centralnim čvorom niti strogo preciznom sinhronizacijom. [5]

U S-MAC protokolu jedan ciklus rada mreže dijeli se na dva vremenska perioda: period aktivnosti (T_{active}) i period spavanja (T_{sleep}). Ovi periodi se izmjenično smjenjuju i imaju isto vrijeme nastupanja za sve čvorove u jednom virtualnom klasteru (čvorovi su potpuno sinhronizirani). Mreža se može sastojati iz više virtualnih klastera. Aktivni period podijeljen je u dva potperioda sastavljena od vremenskih slotova pri čemu se prvi koristi za razmjenu poruka za sinhronizaciju klastera (T_{async}), a drugi se koristi za prenos podataka (T_{adata}). U oba potperioda čvorovi pristupaju mediju koristeći CSMA/CA mehanizam sa slučajnim brojačima koje interno pokreće svaki čvor ukoliko zatiče slobodan medij. Da bi se izbjegao problem skrivenih čvorova u mreži koriste se RTS/CTS kontrolni paketi kojima određeni čvor obavještava druge čvorove o trajanju njegove transmisije. Čvorovi koji prime RTS paket, ažuriraju svoje NAV brojače na vrijeme transmisije čvora koji prenosi i odlaze u stanje spavanja. Na ovaj način se postiže ušteda električne energije.



Slika 1. Vremenski okvir S-MAC protokola

U nastavku su prikazani formati poruka S-MAC protokola. Svaka poruka posjeduje polja LEN, TYPE i CRC. Polje LEN sadrži ukupnu dužinu poruke i omogućava prijemniku da se sinhronizuje i uspješno dekodira poruku. Polje TYPE sadrži oznaku vrste poruke (1 - SYNC, 2 - RTS, 3 - CTS, 4 - ACK, 5 - DATA). Posljednje polje CRC sadrži CRC-16 provjeru pariteta i omogućava otkrivanje grešaka nastalih u procesu prenosa poruke. Na slici 2 prikazan je format sinhronizacijske poruke (SYNC). Osim standardnih polja LEN, TYPE i CRC koje posjeduje svaka poruka, sinhronizacijska poruka sadrži polja SRC, SEQ i SLEEPTIME. Polje SRC i SEQ sadrže adresu pošiljaoca i redni broj poruke, respektivno. Uloga ovih polja je vođenje evidencije o susjednim čvorovima u mreži. Najbitnije polje u SYNC poruci je SLEEPTIME, a ono sadrži vrijeme nakon kojeg će čvor preći u stanje spavanja. Vrijeme spavanja (SLEEPTIME) je

zapisano kao 32-bitni broj sa rezolucijom od 10 ms. Ova veličina je u telekomnikacijama poznata kao *timeticks*.

Dužina LEN	Tip TYPE	Adresa izvora SRC	Broj poruke SEQ	Vrijeme spavanja SLEEPTIME	CRC 16
1B	1B	2B	1B	2B	2B

Slika 2. Format sinhronizacijske poruke (SYNC)

Slika 3 prikazuje format kontrolnih poruka (RTS/CTS/ACK). Ove poruke osim prethodno objašnjenih polja sadrže DST i DUR. Polje DST sadrži adresu primaoca, a polje DUR sadrži očekivano trajanje prenosa skupine poruka RTS, CTS, DATA i ACK, tj. ukupno vrijeme prenosa jednog podatkovnog segmenta. Ovo vrijeme koriste čvorovi kojima nije namijenjen podatkovni paket, kako bi podesili svoje NAV brojače i otišli u stanje spavanja.

Na slici 4 prikazan je format podatkovne poruke (DATA). Osim prethodno objašnjenih polja, ova poruka sadrži polje varijabilne dužine PDU koje nosi podatke protokola višeg sloja (n.pr. mrežni sloj, aplikacijski sloj i sl.). Maksimalna dužina PDU polja je 244 okteta.

Dužina LEN	Tip TYPE	Adresa izvora SRC	Adresa odredišta DST	Trajanje DUR	Broj poruke SEQ	CRC 16
1B	1B	2B	2B	2B	1B	2B

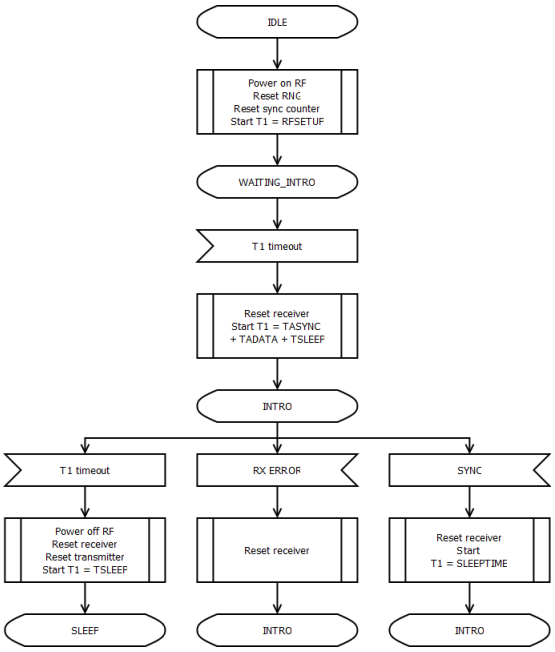
Slika 3. Format kontrolne poruke (RTS/CTS/ACK)

Dužina LEN	Tip TYPE	Adresa izvora SRC	Adresa odredišta DST	Trajanje DUR	Broj poruke SEQ	Podaci PDU	CRC 16
1B	1B	2B	2B	2B	1B	1B-244B	2B

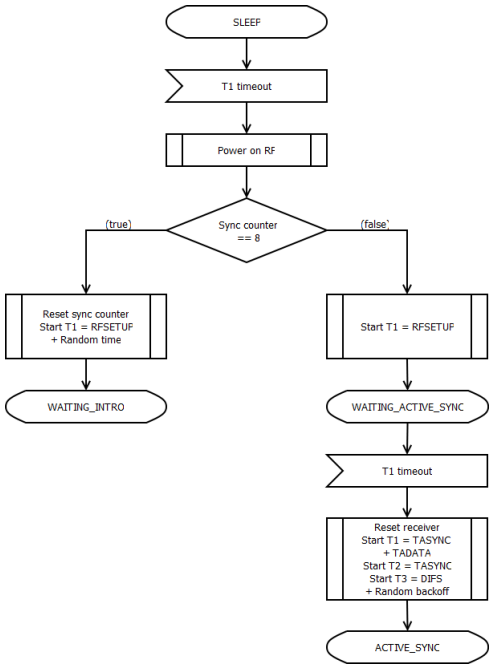
Slika 4. Format podatkovne poruke (DATA)

Za ilustraciju rada protokola koristiti će se SDL dijagrami prikazani u nastavku. U protokolu se koriste tri brojača T1, T2 i T3. Uloga brojača T1 (vanjski brojač) je iniciranje promjene iz stanja u spavanja u aktivno stanje i obrnutno. Uloga brojača T2 (unutrašnji brojač) je iniciranje prelaza u naredno stanje. Brojač T3 se koristi za vremensku odgodu pristupa mediju koju nalaže CSMA/CA mehanizam. Na slici 5 prikazan je SDL dijagram koji opisuje tok događaja od trenutka uključenja bežičnog senzorskog čvora do trenutka odlaska u stanje spavanja. Prilikom uključenja bežični senzorski čvor aktivira RF primopredajnik, resetuje generator slučajnih brojeva i brojač poslanih SYNC paketa, te postavlja brojač T1 na vrijeme potrebno za stabilizaciju i inicijalizaciju RF primopredajnika i odlazi u stanje WAITING_INTRO. Po isteku brojača T1, čvor resetuje prijemnik, postavlja brojač T1 na vrijeme koje odgovara punom periodu (T_{async}+T_{adata}+T_{sleep}) i prelazi u stanje INTRO. U stanju INTRO moguća su tri događaja: istek brojača T1, greška pri prijemu paketa i prijem SYNC paketa. Kada istekne brojač T1, čvor gasi RF primopredajnik, postavlja brojač T1 na vrijeme spavanja i odlazi u stanje SLEEP. U slučaju greške pri prijemu paketa, čvor resetuje prijemnik i ostaju u istom stanju. Pri prijemu SYNC paketa, čvor resetuje prijemnik i postavlja brojač T1 na vrijeme spavanja koje je dobio u SYNC paketu. Na ovaj način, bežični senzorski čvor se sinhronizuje sa postojećim čvorovima u mreži.

Na slici 6 prikazan je proces buđenja bežičnog senzorskog čvora. Po isteku brojača T1, čvor aktivira RF primopredajnik i provjerava da li je broj poslanih SYNC poruka dostigao graničnu vrijednost (u ovom slučaju 8). Brojač poslanih SYNC poruka se koristi za resinhronizaciju čvora koji je migrirao iz jednog klastera u drugi. Ukoliko su periodi jednog i drugog klastera fazno pomjereni, čvor koji je migrirao uvijek bi se budio po rasporedu iz prvog klastera i ne bi mogao razmjenjivati poruke u drugom klasteru. Stoga, ukoliko čvor uzastopno pošalje 8 sinhronizacijskih poruka, a za to vrijeme ne primi niti jednu, to je indicija da je čvor ostao sam u klasteru ili da je migrirao u drugi klaster. U tom slučaju, čvor resetuje brojač poslanih SYNC poruka, podešava brojač T1 na vrijeme stabilizacije RF primopredajnika + neko slučajno vrijeme i prelazi u stanje WAITING_INTRO. Suprotno, čvor podešava brojač T1 na vrijeme stabilizacije RF primopredajnika i prelazi u stanje WAITING_SYNC. Po isteku brojača T1, čvor podešava brojač T1 na vrijeme T_{active}, brojač T2 na vrijeme T_{async} i brojač T3 na DIFS (eng. *DCF interframe space* - dugi period između okvira) + neko slučajno vrijeme, a zatim prelazi u stanje ACTIVE_SYNC.



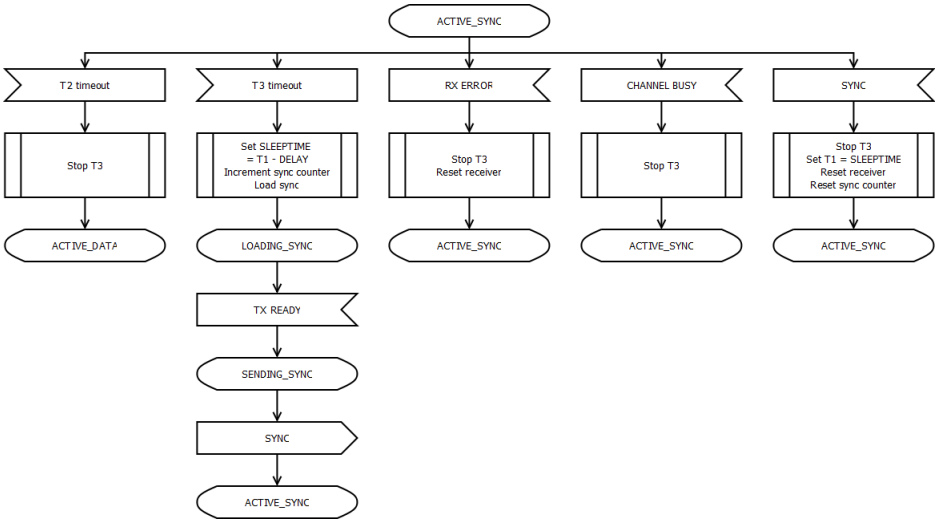
Slika 5. SDL dijagram uvođenja bežičnog senzorskog čvora u mrežu



Slika 6. SDL dijagram buđenja bežičnog senzorskog čvora

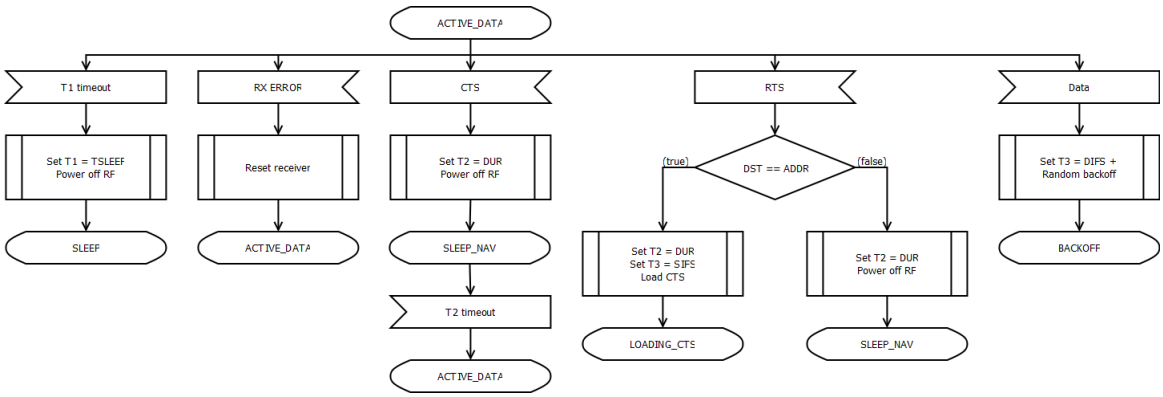
Slika 7 prikazuje sinhronizacijsku fazu bežičnog senzorskog čvora. Ova faza predstavlja prvi dio aktivnog perioda i služi za međusobnu sinhronizaciju čvorova u jednom klasteru. Međusobnom razmjenom vrijednosti brojača T2, čvorovi uklanjaju eventualnu razliku koja je posljedica *jitter*-a lokalnih oscilatora. Trajanje faze je podešeno tako da se može poslati samo jedna sinhronizacijska poruka (SYNC). U ovoj fazi, svi čvorovi se natječu za medij u skladu sa CSMA/CA procedurom. Svaki od čvorova je prethodno pokrenuo brojač T3 sa slučajno odabranom vremenskom zadrškom. Onaj čvor kojem prvo istekne brojač T3 će ući u fazu pripremanja SYNC paketa. U polje SLEEPTIME čvor vrši upis preostalog vremena iz brojača T1 umanjeno za kašnjenje u prenosu SYNC paketa. Zatim inkrementira brojač poslanih SYNC paketa, predajniku signalizira učitavanje paketa i prelazi u stanje LOADING_SYNC. Ukoliko je predajnik spreman, čvor prelazi u stanje SENDING_SYNC i šalje SYNC paket, a zatim se vraća u stanje ACTIVE_SLEEP. Drugi čvorovi će osluškujući medij otkriti da je transmisija u toku i zaustaviti brojač T3. Nakon što prime SYNC paket, ostali čvorovi svoje brojače T1 podešavaju na vrijednost polja

SLEEPTIME, resetuju brojač poslanih SYNC paketa i ostaju u stanju ACTIVE_SYNC. Konačno, na svim čvorovima u klasteru će u isto vrijeme isteći brojač T2 i svi će preći u stanje ACTIVE_DATA.



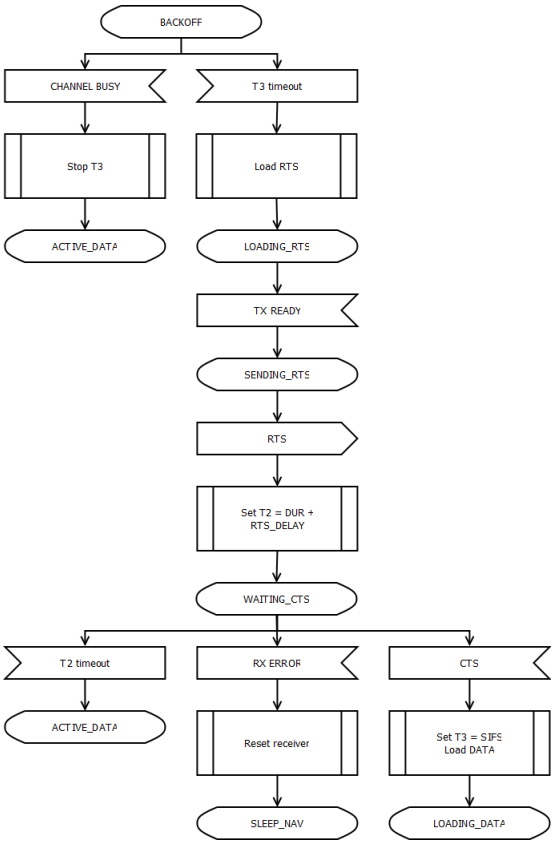
Slika 7. SDL dijagram sinhronizacijske faze

Na slici 8 je prikazana aktivna faza bežičnog senzorskog čvora. Ukoliko čvor dobije od protokola višeg sloja paket koji treba da pošalje (interna poruka *Data*), on podešava brojač T3 na DIFS + neko slučajno vrijeme i prelazi u stanje BACKOFF. Druga mogućnost je prijem poruke RTS. U ovom slučaju čvor provjerava da li je paket adresiran na njega, te ako jeste podešava brojač T2 na DUR, brojač T3 na SIFS, signalizira predajniku da učita CTS paket i prelazi u stanje LOADING_CTS. U suprotnom, čvor podešava brojač T2 na DUR, gasi RF primopredajnik i odlazi u stanje SLEEP_NAV. Treća mogućnost je prijem poruke CTS koji se tretira na isti način kao i prijem RTS poruke adresirane na neki drugi čvor. Konačno, po isteku brojača T1, čvor podešava brojač T1 na T_{sleep} i prelazi u stanje SLEEP.

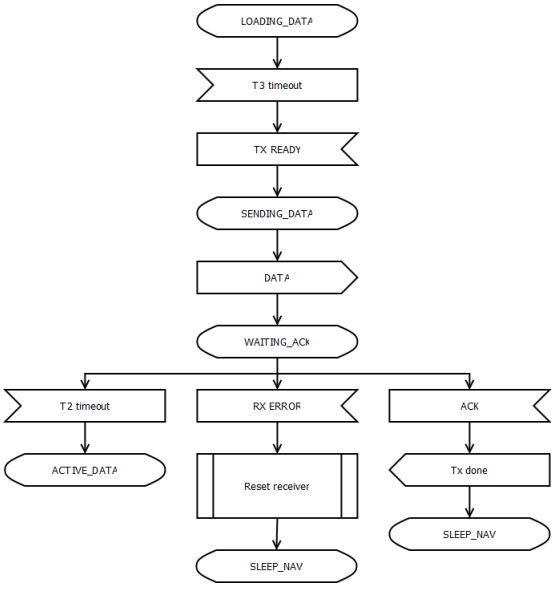


Slika 8. SDL dijagram aktivne faze

Slika 9 prikazuje proceduru slanja RTS i prijema CTS paketa, što predstavlja pripremu za slanje podatkovnog paketa. Ukoliko neki drugi čvor započne prenos podataka, čvor koji se nalazi u stanju BACKOFF će odustati od slanja RTS paketa i preći u stanje ACTIVE_DATA. Ukoliko je medij slobodan i istekne brojač T3, čvor će signalizirati predajniku da učita RTS paket i preći u stanje LOADING_RTS. Ako je predajnik spreman, čvor prelazi u stanje SENDING_RTS i šalje RTS paket. Nakon slanja RTS paketa, čvor podešava brojač T2 na DUR uvećan za kašnjenje u prenosu RTS paketa i prelazi u stanje WAITING_CTS. U ovom stanju, čvor očekuje potvrdu da je primaoc poruke spreman za prijem (CTS paket). Po prijemu CTS paketa, čvor podešava brojač T3 na SIFS, signalizira predajniku da učita DATA paket i prelazi u stanje LOADING_DATA. Ukoliko istekne brojač T2, a čvor nije primio CTS paket, on odustaje od slanja podataka i prelazi u stanje ACTIVE_DATA.



Slika 9. SDL dijagram slanja RTS paketa

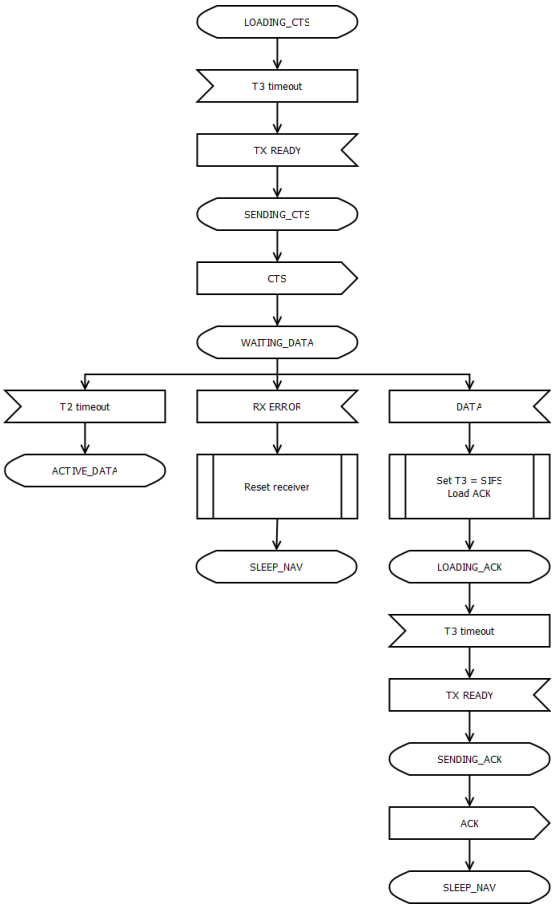


Slika 10. SDL dijagram slanja DATA paketa

Na slici 10 je prikazana prikazana procedura slanja DATA i prijema ACK paketa. Po isteku brojača T3 i ako je predajnik spreman, čvor prelazi u stanje SENDING_DATA i šalje DATA paket, a zatim prelazi u stanje WAITING_ACK. Po prijemu ACK paketa, čvor signalizira protokolu višeg sloja da su podaci uspješno poslani i prelazi u stanje SLEEP_NAV. Ako istekne brojač T2, a čvor nije primio ACK paket, on ne vrši retransmisiju paketa što je uobičajeno za CSMA/CA bazirane protokole, nego prelazi u stanje ACTIVE_DATA. Ova modifikacija je napravljena radi povećanja pravičnosti u mrežama sa kratkim aktivnim periodom (eng. *low duty cycle*).

Slika 11 prikazuje proceduru slanja CTS, prijema DATA i slanja ACK paketa. Kada istekne brojač T3 i ako je predajnik spreman, čvor prelazi u stanje SENDING_CTS i šalje CTS paket, a zatim prelazi

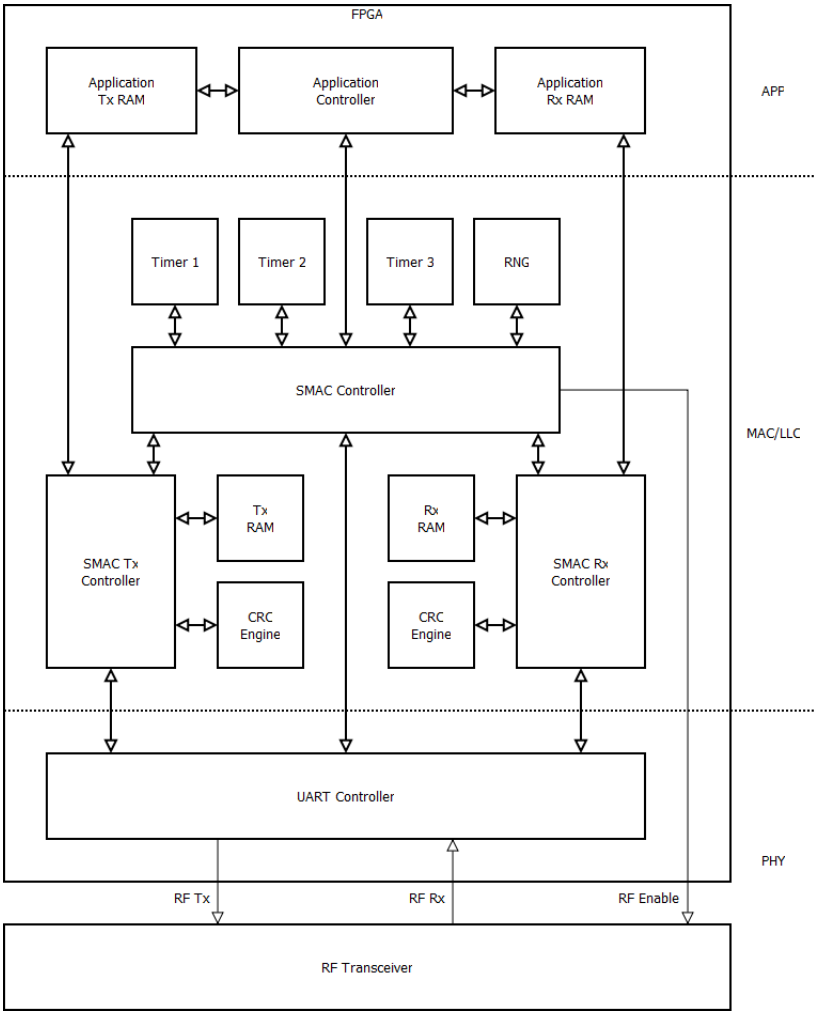
u stanje WAITING_DATA. Ukoliko istekne brojač T2, a čvor ne primi DATA paket, on prelazi u stanje ACTIVE_DATA. Po prijemu DATA paketa, čvor podešava brojač T2 na SIFS, signalizira predajniku da učita ACK paket i prelazi u stanje LOADING_ACK. Kada istekne brojač T3 i ako je predajnik spreman, čvor prelazi u stanje SENDING_ACK i šalje ACK paket. Konačno, čvor prelazi u stanje SLEEP_NAV.



Slika 11. SDL dijagram slanja CTS i ACK paketa

Arhitektura sistema

U ovom radu S-MAC protokol je u potpunosti implementiran na hardveru kao skup međusobno povezanih logičkih cjelina. Na slici 12 je prikazana blokovska organizacija arhitekure sistema. Blokovi su podijeljeni u tri sloja prema OSI/ISO modelu i na taj način je naznačeno šta pripada fizičkom sloju, šta sloju linka podataka, a šta aplikacijskom sloju. Osim toga, blokovi su u implementaciji hijerarhijski organizirani prema referentnom OSI/ISO modelu.



Slika 12. Arhitektura sistema

Fizički sloj čine RF primopredajnik zajedno sa UART kontrolerom koji je implementiran na FPGA. Uloga UART kontrolera u ovom dizajnu je dvojaka: paralelno-serijska i serijsko-paralelna konverzija toka podataka, te serijski interfejs prema RF primopredajniku. Ukoliko se odlučimo za promjenu vrste RF primopredajnika i njegovog interfejsa prema FPGA, dovoljno je UART kontroler zamijeniti adekvatnim kontrolerom (n.pr. SPI, I2C i dr.). Na sloju linka podataka se nalazi implementacija S-MAC protokola. S-MAC protokol se sastoji od sljedećih logičkih cjelina:

- *SMAC Tx Controller* - upravljačka jedinica za slanje podataka (predajnik),

- *Tx RAM* - memorijski spremnik podataka spremnih za slanje,
- *SMAC Rx Controller* - upravljačka jedinica za prijem podataka (prijemnik),
- *Rx RAM* - memorijski spremnik primljenih podataka,
- *CRC Engine* - generator CRC-16 bita za provjeru pariteta, tj. detekciju grešaka nastalih u prenosu podataka,
- *SMAC Controller* - upravljačka jedinica S-MAC protokola,
- *Timer 1, 2 i 3* - brojači,
- *RNG* - generator slučajnih brojeva baziran na tzv. *ring* oscilatoru.

Na aplikacijskom sloju se nalazi implementacija probne aplikacije namijenjene za testiranje S-MAC protokola. Aplikacija se sastoji od:

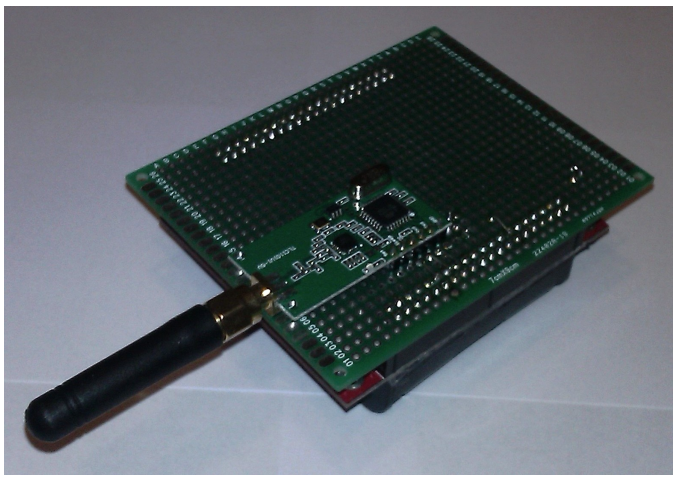
- *Application Controller* - upravljačka jedinica aplikacije,
- *Application Tx RAM* - memorijski spremnik podataka spremnih za slanje,
- *Application Rx RAM* - memorijski spremnik primljenih podataka.

Platforma za implementaciju

Za implementaciju S-MAC protokola korištene su sljedeće hardverske komponente:

- Altera FPGA EP2C5T144 razvojni sistem,
- RF primopredajnik TLC1101V1-5V.

Na slici 13 prikazan je bežični senzorski čvor napravljen da služi kao platforma za implementaciju S-MAC protokola.

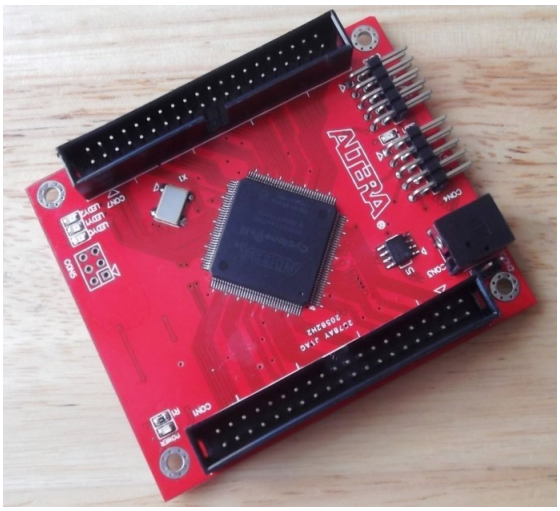


Slika 13. Platforma za implementaciju S-MAC protokola

Razvojni sistem (slika 14) korišten u ovom radu sadrži FPGA čip EP2C5T144, čip za serijsku konfiguraciju EPCS4 sa flash memorijom kapaciteta 4 Mbita, kvarcni oscilator od 50 MHz, 3 LED-ice i elektroniku za napajanje. Napon napajanja FPGA čipa je 1,8 V, a maksimalna izlazna struja po pinu iznosi 40 mA. Napajanje FPGA čipa i 3.3-V LVTTL periferije je osigurano regulatorima napona AZ1084D-ADJE1 i AZ1085D-3.3E1. Programiranje je moguće korištenjem JTAG ili Active Serial interfejsa. Preko JTAG interfejsa se vrši direktno programiranje SRAM ćelija FPGA čipa i može biti korisno u fazi razvoja i testiranja dizajna. Za trajno pohranjivanje dizajna koristi se Active Serial interfejs preko kojeg se programira sadržaj čipa za serijsku konfiguraciju EPCS4, a koji nakon uključivanja napajanja vrši konfiguraciju FPGA čipa. Na štampanoj ploči razvojnog sistema nalaze se dva 40-pinska konektora čime je omogućena modularnost u razvoju prototipa ili gotovog rješenja.

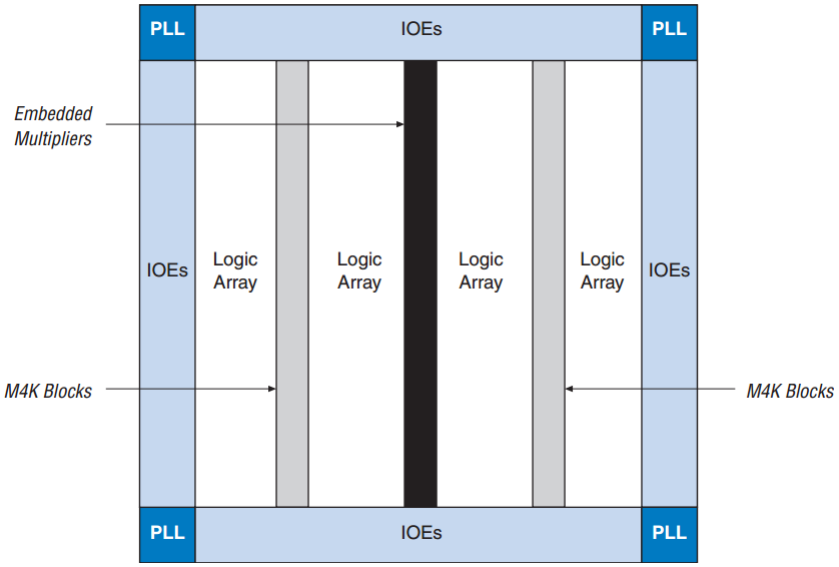
Altera Cyclone II EP2C5T144 FPGA čip karakterišu relativno niska cijena, te 60% bolje performanse i upola manja potrošnja električne energije od 90-nm prethodnika. Niska cijena i optimizirani skup mogućnosti čine ovaj čip idealnim za upotrebu u automobilskoj industriji, potrošačkoj elektronici, komunikacijama, obradi video signala, testiranju, mjerenju i sl. Korišteni čip posjeduje sljedeće karakteristike:

- 4608 logičkih elemenata,
- 26 M4K ugrađenih memorijskih blokova (ukupno 119808 memorijskih bita),
- 13 ugrađenih množača formata 18x18,
- 2 PLL-a,
- 89 ulazno/izlaznih pinova.



Slika 14. Altera FPGA EP2C5T144 razvojni sistem

Cyclone II čipovi imaju dvodimenzionalnu arhitekturu za implementaciju željene logike (slika 15). Međusobne veze redova i kolona u izvedbama sa različitim brzinama omogućavaju povezivanje nizova logičkih blokova (eng. *Logic Array Block*, LAB), ugrađenih memorijskih blokova i ugrađenih množača. Logički niz se sastoji iz više LAB-ova, a svaki LAB sadrži 16 logičkih elemenata (eng. *Logic Element*, LE). LE je mala jedinica koja omogućava efikasnu implementaciju korisničkih logičkih funkcija. LAB-ovi su grupisani u redove i kolone preko cijele površine čipa.



Slika 15. Arhitektura Cyclone II FPGA čipa [7]

Čip podržava mrežu globalnih taktova sa dvije fazno zaključane petlje (eng. *Phase-Locked Loop*, PLL). Mreža globalnih taktova se sastoji od 8 linija za prenos takt signala kroz cijeli čip, a pruža takt signal do svih resursa unutar čipa kao što su ulazno-izlazni elementi, LE-ovi, ugrađeni množači i ugrađeni memorijski blokovi. Linije za prenos takt signala se također mogu koristiti za opću namjenu tamo gdje je izlazni faktor opteretivosti jako visok. PLL-ovi omogućavaju generisanje takt signala sa željenom frekvencijom i faznim pomakom, koji se također može koristiti kao podrška brzoj diferencijalnoj ulazno-izlaznoj komunikaciji.

M4K memorijski blokovi su dvo-portni memorijski blokovi sa 4 Kb memorije plus paritetni biti (4608 bita). Ovi blokovi se mogu mapirati u dvoportne, jednostavne dvoportne i jednoportne memorije sa do 36-bita širokom sabirnicom, a maksimalna frekvencija na kojoj mogu raditi je 260 MHz. Memorijski blokovi su organizovani u dvije kolone i nalaze se između nekih LAB-ova.

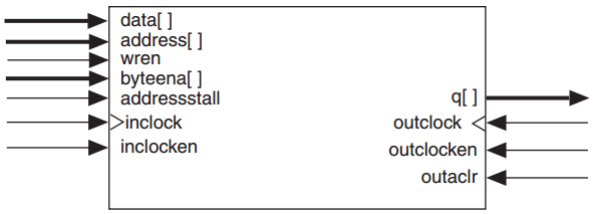
M4K memorijski blokovi sadrže ulazne registre pomoću kojih se sinhronizuje upis i izlazne registre za pipeline podatkaka, čime se poboljšavaju performanse sistema. Svi memorijski blokovi su u potpunosti

sinhroni, što znači da se ulazni signali dovode preko registara, a izlazni podaci mogu opcionalno prolaziti kroz registar. M4K blokovi podržavaju sljedeće režime rada:

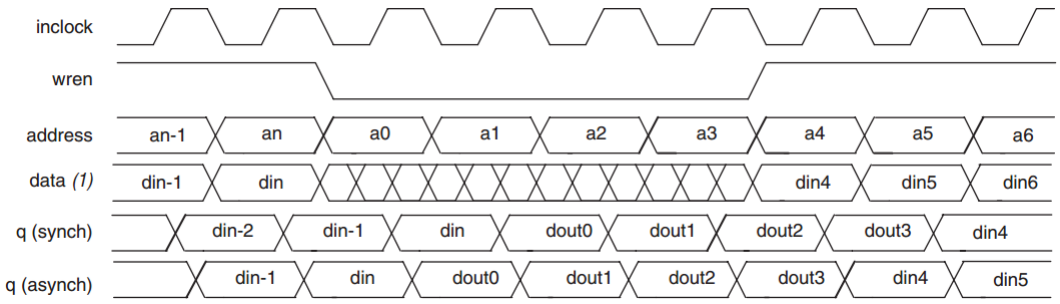
- jednoportni,
- jednostavni dvoportni,
- pravi dvo-portni (bidirekcioni dvo-portni),
- pomjerački registar,
- ROM,
- FIFO međuspremnik.

U ovom radu, memorijski blokovi su korišteni u dva režima rada: jednoportni za realizaciju memorijskog spremnika za prijem i predaju podataka (*Tx RAM* i *Rx RAM*), jednostavni dvoportni za realizaciju aplikacijske memorije (*Application Tx RAM* i *Application Rx RAM*).

Jednoportni režim rada podržava nesimulatne operacije čitanja i pisanja. Slika 16 prikazuje jednoportnu konfiguraciju Cyclone II memorijskog bloka. Na slici 17 je prikazan vremenski dijagram čitanja i pisanja u jednoportnom režimu. U jednoportnom režimu, izlazi su u čitanje-tokom-pisanja režimu, što znači da podaci koji se upisuju prolaze kroz memoriju prema izlazu. Ako se izlazni registar ne koristi, novi podaci su dostupni na rastućoj ivici takt signala u istom takt intervalu u kojem su upisani. U jednoportnom režimu moguće su sljedeće konfiguracije sabirnice: 4K x 1, 2K x 2, 1K x 4, 512 x 8, 512 x 9 (paritetni biti se koriste kao informacijski), 256 x 16, 256 x 18 (paritetni biti se koriste kao informacijski), 128 x 32, 128 x 36 (paritetni biti se koriste kao informacijski). Memorijski spremnici za prijem i predaju podataka su realizirani u 512 x 8 konfiguraciji.



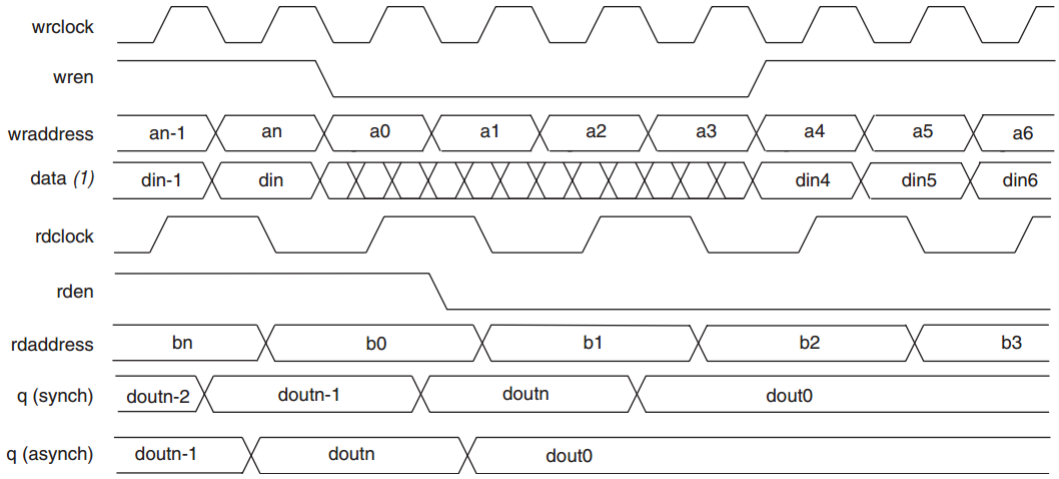
Slika 16. Jednoportna konfiguracija Cyclone II memorijskog bloka [7]



Slika 17. Vremenski dijagram u jednoportnom radnom režimu [7]

Jednostavni dvoportni režim rada podržava simultane opracije čitanja i pisanja, ali sa istim signalom takta. Slika 18 prikazuje jednostavnu dvoportnu konfiguraciju Cyclone II memorijskog bloka. Na slici 19 je prikazan vremenski dijagram čitanja i pisanja u jednostavnom dvoportnom režimu. U jednostavnom dvoportnom režimu podržane su različite kombinacije konfiguracija sabirnica za čitanje i pisanje. Aplikacijska memorija je realizirana u 512 x 8 konfiguraciji na obje sabirnice. U jednostavnom dvoportnom režimu, memorijski blokovi imaju jedan signal za omogućavanje upisa i jedan za omogućavanja čitanja. Kada je signal za omogućavanje čitanja deaktiviran, trenutni podaci ostaju na izlaznim portovima. Ako se signal za omogućavanje čitanja aktivira tokom operacije pisanja sa istom adresnom lokacijom, izlaz drži stare podatke pohranjene na toj memorijskoj adresi.

Slika 18. Jednostavna dvoportna konfiguracija Cyclone II memorijskog bloka [7]

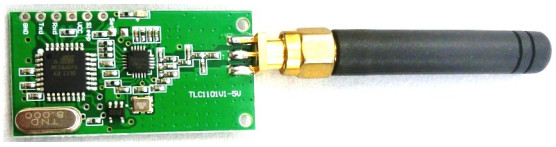


Slika 19. Vremenski dijagram u jednostavnom dvoportnom radnom režimu [7]

Svaki ugrađeni množač omogućava implementaciju dva množača formata 9x9 bita ili jedan množač formata 18x18 bita, a maksimalna frekvencija na kojoj mogu raditi je 250 MHz. Ugrađeni množači su organizovani u jednoj koloni na sredini čipa.

Svaki pin Cyclone II čipa je spojen na ulazno-izlazni element (eng. *Input/Oputput Element*, IOE). IOE-ovi se nalaze na svim krajevima čipa, a podržavaju različite jednostruko vezane i diferencijalne ulazno-izlaze standarde. Svaki IOE sadrži bidirekcioni ulazno-izlazni međuspremnik i tri registra za registrovanje ulaza, izlaza i omogućen-izlaz signala. U ovom radu, ulazno-izlazni elementi su konfigurisani da podržavaju 3.3-V LVTTL standard. LVTTL standard specificira širok opseg ulaznog napona $-0,3V \leq V_I \leq 3,9V$, ali Altera preporučuje opseg $-0,5V \leq V_I \leq 4,1V$.

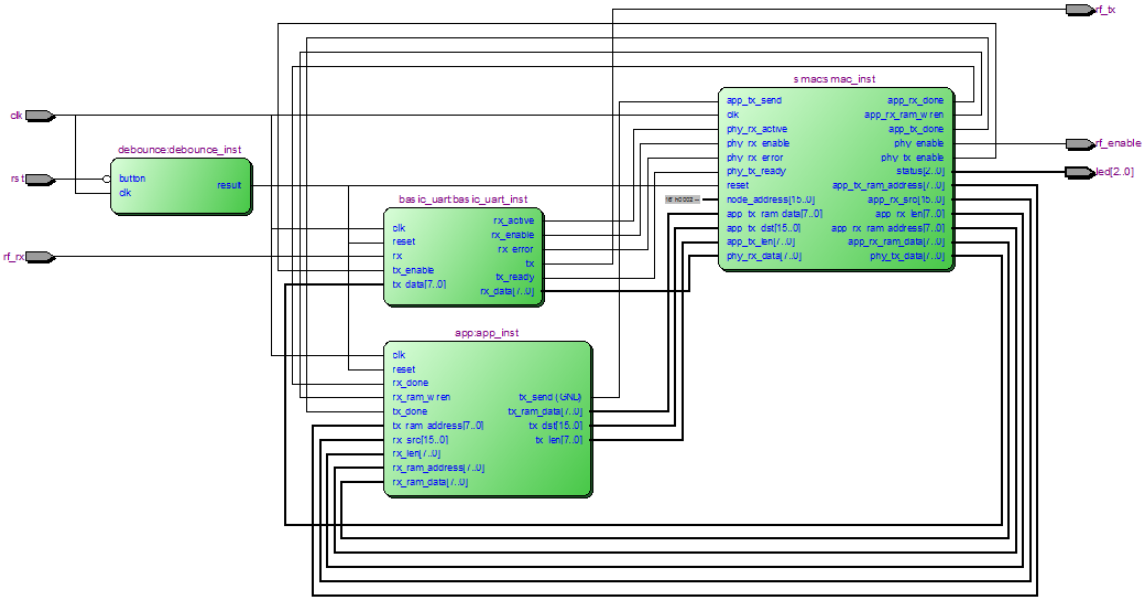
Za razmjenu informacija između bežičnih senzorskih čvorova korišten je RF primopredajni modul baziran na CC1101 RF čipu (slika 20). RF modul prenose podatke na frekvenciji 433 MHz sa maksimalnom predajnom snagom od 10 mW. Između FGPA čipa i RF modula korišten je UART komunikacijski interfejs sa naponskim nivoima prema 3.3-V LVTTL standardu. Komunikacijski modul koristi FSK modulacijski metod za prenos informacija na fizičkom sloju referentnog OSI/ISO modela. CC1101 također pruža i zaštitu protiv ometanja, te omogućava prenos informacija na 256 nepreklapajućih kanala širine 200 kHz. U ovom radu, komunikacijski modul je konfigurisan tako da podržava prenos informacija brzinom 9600 bps na maksimalnoj udaljenosti od 200 m. Masa RF primopredajnika je vezan na kolektor NPN tranzistora BC547. Emiter tranzistora je vezan na masu, a baza na jedan od izlaznih pinova FPGA čipa. Na ovaj način se iz FPGA čipa upravlja napajanjem RF primopredajnika.



Slika 20. RF komunikacijski modul TLC1101V1-5V

Realizacija komponenti sistema

U nastavku je dat pregled komponenti sistema realiziranih upotrebom VHDL jezika. U nekim slučajevima će radi bolje ilustracije biti prikazani dijelovi VHDL koda, a kompletni kodovi su dati u prilogu ovog dokumenta. Svi kodovi su pisani prema VHDL 2008 standardu. Komponente su organizovane hijerarhijski. Na slici 21 je dat šematski prikaz najvišeg nivoa hijerarhije.



Slika 21. Šematski prikaz najvišeg nivoa hijerarhije

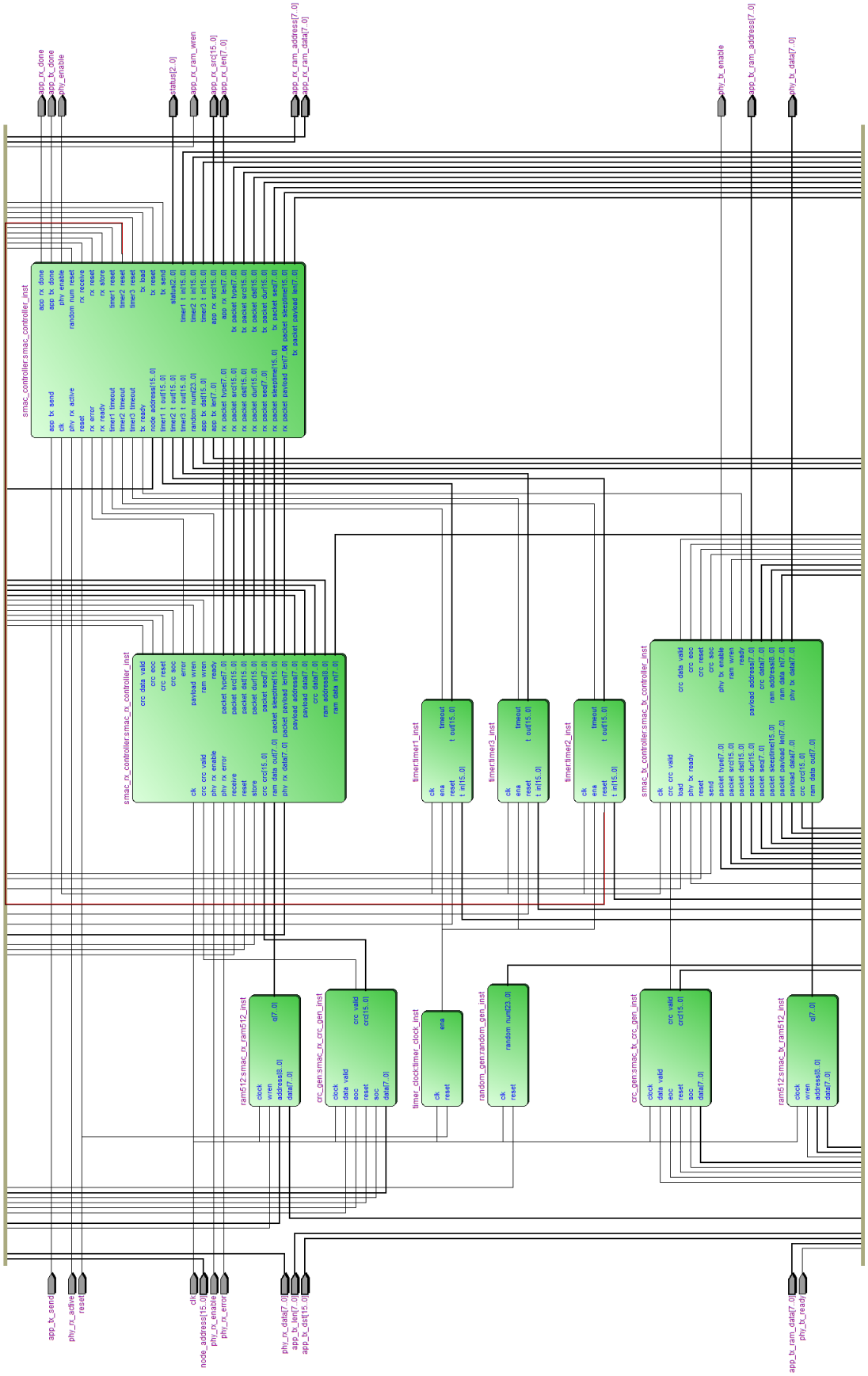
Najviši nivo hijerarhije posjeduje sljedeće ulazne signale:

- *clk* - signal takta (mapiran na pin na koji je spojen kvarcni oscilator),
- *rst* - asinhroni reset signal (mapiran na taster na razvojnom sistemu, aktivan u stanju logičke "0"),
- *rf_rx* - serijski ulaz za podatke (mapiran na pin na koji spojen izlaz iz RF primopredajnika),

te izlazne signale:

- *rx_tx* - serijski izlaz za podatke (mapiran na pin na koji je spojen ulaz u RF primopredajnik),
- *rf_enable* - signal za omogućavanje rada RF primopredajnika (mapiran na pin na koji je spojena baza tranzistora),
- *led[2..0]* - statusni signal (mapiran na 3 LED-ice na razvojnom sistemu).

Na slici 22 je dat šematski prikaz prvog nižeg nivoa hijerarhije na kojem su implementirane sve komponente S-MAC protokola.



Slika 21. Šematski prikaz S-MAC protokola

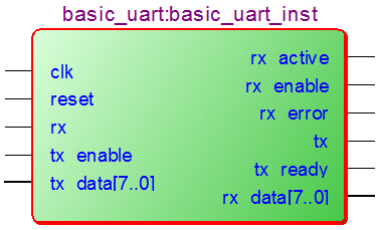
UART Controller

Komponenta UART Controller izvršava dvije funkcije: paralelno-serijska i serijsko-paralelna konverziju toka podataka, te serijski interfejs prema RF primopredajniku. Na slici 21 je dat šematski prikaz komponente UART Controller. Komponenta posjeduje sljedeće ulazne signale:

- *clk* - signal takta,
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),
- *rx* - serijski ulaz za prijem podataka (spojen na izlaz iz RF primopredajnika),
- *tx_enable* - signal za omogućavanje slanja podataka (aktivan u stanju logičke "1"),
- *tx_data[7..0]* - podaci za slanje (8 bita = 1 oktet),

te izlazne signale:

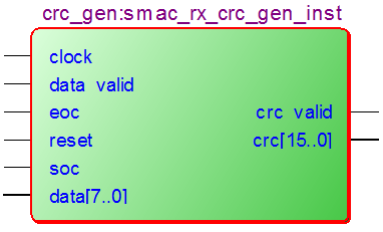
- *rx_active* - signalizira da je prijem podataka u toku (aktivan u stanju logičke "1"),
- *rx_enable* - signalizira da su podaci uspješno primljeni (aktivan u stanju logičke "1"),
- *rx_error* - signalizira grešku pri prijemu podataka, tj. izostanak stop bita (aktivan u stanju logičke "1"),
- *tx* - serijski izlaz za slanje podataka (spojen na ulaz RF primopredajnika),
- *tx_ready* - signalizira da je predajnik spreman za slanje podataka (aktivan u stanju logičke "1"),
- *rx_data[7..0]* - primljeni podaci (8 bita = 1 oktet).



Slika 21. Komponenta UART Controller

CRC Engine

Komponenta CRC Engine je zadužena za generisanje bita za provjeru pariteta prema CRC-16-CCITT standardu. Upotrebom bita za provjeru pariteta može se otkriti greška u prenosu podataka i eventualno obaviti retransmisija. Standard CRC-16-CCITT podrazumijeva upotrebu generatorskog polinoma $x^{16} + x^{12} + x^5 + 1$, a poznat je po upotrebi u velikom broju komunikacijskih tehnologija (X.25, V.41, CDMA, Bluetooth, XMODEM, HDLC, PPP, IrDA, itd.). VHDL kod za ovu komponentu je generisan upotrebom Web aplikacije http://www.electronicdesignworks.com/utilities/crc_generator/crc_generator.htm. Na slici 22 je dat šematski prikaz komponente CRCEngine.



Slika 22. Komponenta CRC Engine

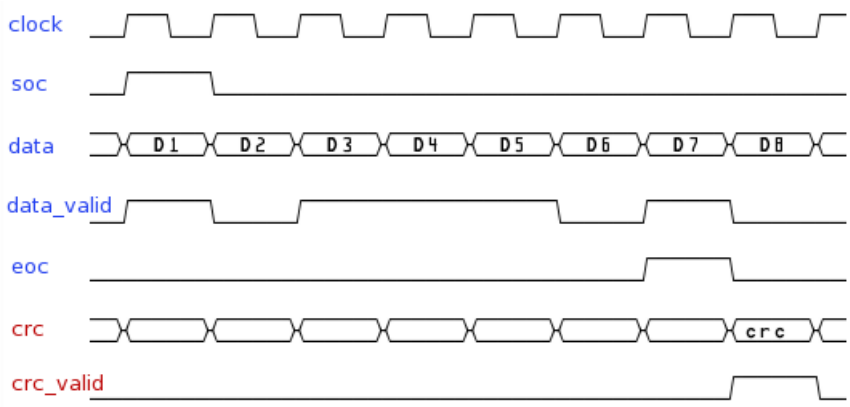
Komponenta posjeduje sljedeće ulazne signale:

- *clock* - signal takta,
- *data_valid* - signal za omogućavanje proračuna CRC-a nad ulaznim podacima (aktivan u stanju logičke "1"),
- *eoc* - indikacija zadnjeg okteta nad kojim se vrši proračun CRC-a (aktivan u stanju logičke "1"),
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),
- *soc* - indikacija prvog okteta nad kojim se vrši proračun CRC-a (aktivan u stanju logičke "1"),
- *data[7..0]* - podaci nad kojima se vrši proračun CRC-a (8 bita = 1 oktet),

te izlazne signale:

- *crc_valid* - signalizira da je proračun CRC-a završen (aktivan u stanju logičke "1"),
- *crc[15..0]* - proračunati CRC (16 bita = 2 okteta).

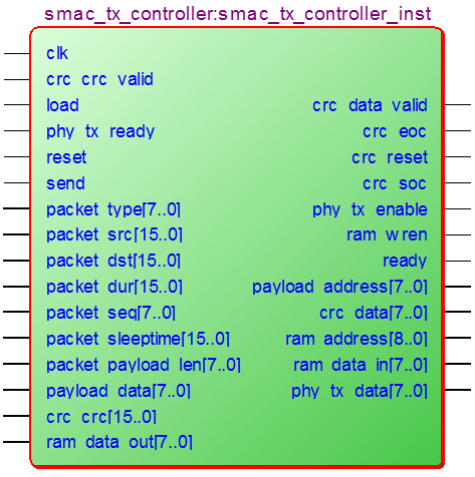
Na slici 23 je prikazan vremenski dijagram komponente CRC Engine.



Slika 23. Vremenski dijagram komponente CRC Engine [8]

SMAC Tx Controller

Komponenta SMAC Tx Controller predstavlja upravljačku jedinicu za slanje podataka. Zadužena je za sastavljanje okvira poruke (kodiranje) u zavisnosti od tipa poruke, fragmentaciju i slanje okvira. Na slici 24 je dat šematski prikaz komponente.



Slika 24. Komponenta SMAC Tx Controller

Komponenta posjeduje sljedeće ulazne signale:

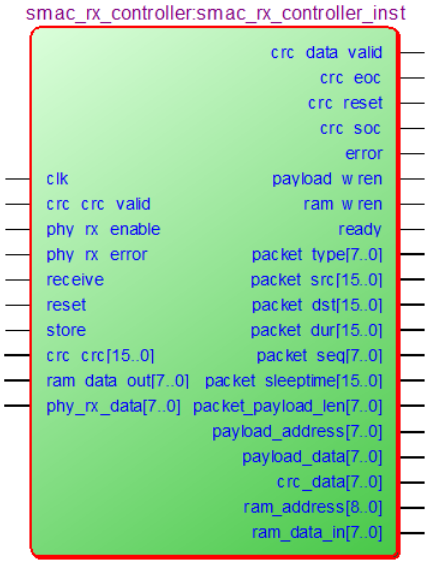
- *clk* - signal takta,
- *crc_crc_valid* - validacija izlaza iz komponente CRC Engine (aktivan u stanju logičke "1"),
- *load* - signal za početak kodiranja okvira poruke i učitavanja u predajnu memoriju (aktivan u stanju logičke "1"),
- *phy_tx_ready* - signal koji potvrđuje da je UART Controller spreman za slanje podataka (aktivan u stanju logičke "1"),
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),
- *send* - signal za početak slanja spremljenog okvira (aktivan u stanju logičke "1"),
- *packet_type[7..0]* - tip paketa koji se šalje (8 bita = 1 oktet),
- *packet_src[15..0]* - izvorišna adresa paketa koji se šalje (16 bita = 2 okteta),
- *packet_dst[15..0]* - odredišna adresa paketa koji se šalje (16 bita = 2 okteta),
- *packet_dur[15..0]* - trajanje transmisije paketa (16 bita = 2 okteta),
- *packet_seq[7..0]* - redni broj paketa koji se šalje (8 bita = 1 oktet),
- *packet_sleeptime[15..0]* - vrijeme spavanja čvora koji šalje paket (16 bita = 2 okteta),
- *packet_payload_len[7..0]* - dužina aplikacijskog dijela paketa koji se šalje (8 bita = 1 oktet),
- *crc_crc[15..0]* - proračunati CRC (16 bita = 2 okteta),
- *ram_data_out[7..0]* - podaci koji se čitaju iz predajne memorije (8 bita = 1 oktet),

te izlazne signale:

- *crc_data_valid* - validacija podataka za koje se računa CRC (aktivan u stanju logičke "1"),
- *crc_eoc* - validacija kraja okvira za koji se računa CRC (aktivan u stanju logičke "1"),
- *crc_reset* - asinhroni reset komponente CRC Engine (aktivan u stanju logičke "1"),
- *crc_soc* - validacija početka okvira za koji se računa CRC (aktivan u stanju logičke "1"),
- *phy_tx_enable* - signal za omogućavanje slanja okteta preko komponente UART Controller (aktivan u stanju logičke "1"),
- *ram_wren* - signal za omogućavanje upisa u predajnu memoriju,
- *ready* - signal kojim komponenta potvrđuje da je spremna za učitavanje ili slanje podataka (aktivan u stanju logičke "1"),
- *payload_address[7..0]* - memorijska adresa za podatak koji se čita iz aplikacijske predajne memorije (8 bita => 256 lokacija),
- *crc_data[7..0]* - podaci za koje se računa CRC (8 bita = 1 oktet),
- *ram_address[8..0]* - memorijska adresa za podatak koji se upisuje u predajnu memoriju (9 bita => 512 lokacija),
- *ram_data_in[7..0]* - podatak koji se upisuje u predajnu memoriju (8 bita = 1 oktet),
- *phy_tx_data[7..0]* - podatak koji se šalje pomoću komponente UART Controller (8 bita = 1 oktet).

SMAC Rx Controller

Komponenta SMAC Tx Controller predstavlja upravljačku jedinicu za prijem podataka. Zadužena je za prijem okvira i dekodiranje poruke. Na slici 25 je dat šematski prikaz komponente.



Slika 25. Komponenta SMAC Rx Controller

Komponenta posjeduje sljedeće ulazne signale:

- *clock* - signal takta,
- *crc_crc_valid* - validacija proračunatog CRC-a (aktivan u stanju logičke "1"),
- *phy_rx_enable* - signal koji potvrđuje da je UART Controller uspješno primio podatak (aktivan u stanju logičke "1"),
- *phy_rx_error* - signal koji indicira da je došlo do greške pri prijemu podataka u UART Controller-u (aktivan u stanju logičke "1"),
- *receive* - signal za početak prijema i dekodiranja okvira (aktivan u stanju logičke "1"),
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),
- *store* - signal za početak spremanja aplikacijskog dijela primljenog okvira u aplikacijsku memoriju (aktivan u stanju logičke "1"),
- *crc_crc[15..0]* - proračunati CRC (16 bita = 2 okteta),
- *ram_data_out[7..0]* - podatak koji se čita iz prijemne memorije (8 bita = 1 oktet),
- *phy_rx_data[7..0]* - podatak koji je primio UART Controller (8 bita = 1 oktet),

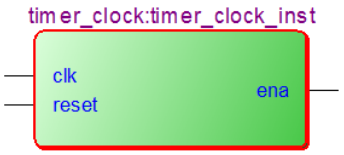
te izlazne signale:

- *crc_data_valid* - validacija podataka za koje se računa CRC (aktivan u stanju logičke "1"),
- *crc_eoc* - validacija kraja okvira za koji se računa CRC (aktivan u stanju logičke "1"),
- *crc_reset* - asinhroni reset komponente CRC Engine (aktivan u stanju logičke "1"),
- *crc_soc* - validacija početka okvira za koji se računa CRC (aktivan u stanju logičke "1"),
- *error* - signalizira da je došlo do greške pri prijemu okvira (aktivan u stanju logičke "1"),
- *payload_wren* - signal za omogućavanje upisa u aplikacijsku memoriju (aktivan u stanju logičke "1"),

- *ram_wren* - signal za omogućavanje upisa u prijemnu memoriju (aktivan u stanju logičke "1"),
- *ready* - signalizira da je kontroler spreman za prijem okvira ili spremanje aplikacijskog dijela okvira u aplikacijsku memoriju (aktivan u stanju logičke "1"),
- *packet_type[7..0]* - tip paketa koji je primljen (8 bita = 1 oktet),
- *packet_src[15..0]* - izvorišna adresa paketa koji je primljen (16 bita = 2 okteta),
- *packet_dst[15..0]* - odredišna adresa paketa koji je primljen (16 bita = 2 okteta),
- *packet_dur[15..0]* - trajanje transmisije paketa (16 bita = 2 okteta),
- *packet_seq[7..0]* - redni broj paketa koji je primljen (8 bita = 1 oktet),
- *packet_sleeptime[15..0]* - vrijeme spavanja čvora koji se poslao paket (16 bita = 2 okteta),
- *packet_payload_len[7..0]* - dužina aplikacijskog dijela paketa koji je primljen (8 bita = 1 oktet),
- *payload_address[7..0]* - memorijska adresa za podatak koji se upisuje u aplikacijsku prijemnu memoriju (8 bita => 256 lokacija),
- *payload_data[7..0]* - podatak koji se upisuje u aplikacijsku prijemnu memoriju (8 bita = 1 oktet),
- *crc_data[7..0]* - podaci za koje se računa CRC (8 bita = 1 oktet),
- *ram_address[8..0]* - memorijska adresa za podatak koji se upisuje u prijemnu memoriju (9 bita => 512 lokacija),
- *ram_data_in[7..0]* - podatak koji se upisuje u prijemnu memoriju (8 bita = 1 oktet).

Timer

Komponenta Timer predstavlja brojač koji broji od zadanog broja prema nuli. S obzirom da je osnovni takt signal ima frekvenciju 50 MHz, a brojači rezoluciju 10 ms, realizirana je pomoćna komponenta koja ima zadatak da frekvenciju osnovnog takt signala podijeli sa 500000. Na slici 26 je dat šematski prikaz djeljitelja takta. Takt signal se kroz FPGA čip prostire mrežom globalnih taktova, tj. linijama koje imaju najmanje moguće kašnjenje, pa klasično dijeljenje frekvencije takt signala može biti dovesti do problema sa sinhronizacijom. Umjesto toga, sklop generiše omogućavajući signal na svakih 500000 impulsa osnovnog takt signala (izlazni signal *ena*).



Slika 26. Djeljitelj frekvencije takt signala faktorom 50000

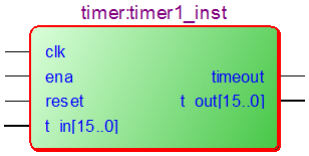
Na slici 27 je dat šematski prikaz komponente Timer. Komponenta Timer posjeduje sljedeće ulazne signale:

- *clk* - signal takta,
- *ena* - signal za omogućavanje brojača (aktivan u stanju logičke "1"),
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),
- *t_in[15..0]* - vrijednost na koju se postavlja brojač (za isključenje brojača postavlja se vrijednost 65535),

te izlazne signale:

- *timeout* - signalizira da je brojač došao na vrijednost 0 (aktivan u stanju logičke "1"),

- *t_ out[15..0]* - trenutna vrijednost brojača, tj. preostalo vrijeme (16 bita, za rezoluciju 10 ms maksimalna vrijednost iznosi 655,34 sekundi).



Slika 27. Komponenta Timer

RNG

Komponenta RNG predstavlja generator slučajnih brojeva. Iako se u digitalnim sistemima preporučuje upotreba pseudo-slučajnih generatora brojeva (eng. *Pseudo-Random Number Generator*, PRNG) koji su bazirani na pomičnom registru, taj pristup ovdje nije korišten. Naime, događaj odabira slučajnog broja pokreće komponenta Timer, pa bi u slučaju upotrebe PRNG-a došlo do sinhronizacije sa periodom sekvence i rezultujući niz slučajnih brojeva bi izgubio osobinu "slučajnosti". Zbog toga je u ovom radu RNG realiziran pomoću tzv. *ring* oscilatora. *Ring* oscilatori se dobiju kada se namjerno ili slučajno napravi kombinatorna petlja, tj. na ulaz neke logičke funkcije se dovodi njen izlaz. Alati za sintezu VHDL koda otkrivaju kombinatorne petlje i u tim situacijama na izlaz logičke funkcije postavljaju *latch* kolo. Zbog toga je umjesto običnog *buffer* signala korištena fizička logička ćelija LCELL koja ima kašnjenje ovisno o temperaturi FPGA čipa. Ako se LCELL upotrijebi u realizaciji *ring* oscilatora, frekvencija signala koji generiše oscilator će ovisiti o temperaturi. Istraživanja su pokazala da se zbog prirode termičkog procesa, *jitter* dobijenog signala podvrgava Gaussovoj raspodjeli. Ukoliko uzorkujemo dobijeni signal sa preciznim taktom (kvarcni oscilator), vjerovatnoća da ćemo "pogoditi" logičku "1" ili "0" se podvrgava Gaussovoj raspodjeli. Sabiranjem izlaza iz nekoliko *ring* oscilatora dobije se signal čiji *jitter* ima veću standardnu devijaciju, pa se na taj način povećava kvalitet generisanog slučajnog broja. VHDL kod koji opisuje RNG komponentu je:

```
LIBRARY IEEE;
LIBRARY ALTERA;
USE IEEE.STD_LOGIC_1164.ALL;
USE ALTERA.ALL;

ENTITY random_gen IS

    PORT ( clk : IN STD_LOGIC;

        reset: IN STD_LOGIC;
        random_num : OUT STD_LOGIC_VECTOR(23 DOWNTO 0) );

END random_gen;
ARCHITECTURE arch_random_gen OF random_gen IS

    SIGNAL x1, x2, x3: STD_LOGIC;
    SIGNAL rand_temp: STD_LOGIC_VECTOR(23 DOWNTO 0);
    COMPONENT LCELL

        PORT (a_in : IN STD_LOGIC;
        a_out : OUT STD_LOGIC);

    END COMPONENT;

BEGIN

    -- Ring oscillator 1
    lcell_inst1: LCELL
    PORT MAP ( a_in => NOT(x1), a_out => x1);
    -- Ring oscillator 2
    lcell_inst2: LCELL
    PORT MAP ( a_in => NOT(x2), a_out => x2);
    -- Ring oscillator 3
```

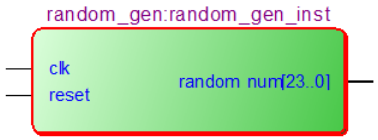
```
    lcell_inst3: LCELL
    PORT MAP ( a_in => NOT(x3), a_out => x3);

    -- Sampler with shift register
    PROCESS (clk)
    BEGIN
        IF reset = '1' THEN
            rand_temp(23 DOWNTO 1) <= (OTHERS => '0');
            rand_temp(0) <= '1';
        ELSIF RISING_EDGE(clk) THEN

            rand_temp <= rand_temp(22 DOWNTO 0) & (x1 XOR x2 XOR x3);
        END IF;
        random_num <= rand_temp;
    END PROCESS;

END arch_random_gen;
```

Komponentu čine 3 *ring* oscilatora i sklop za odabiranje sa pomičnim registrom. Signali koji se odabiraju na izlazu sva tri ring oscilatora se sabiraju pomoću XOR logičke funkcije i rezultat se smiješta u 24-bitni pomični registar. Izlaz iz komponente je sadržaj pomičnog registra. Na slici 28 je dat šematski prikaz komponente RNG. Komponenta posjeduje sljedeće ulazne signale:

- *clk* - signal takta,
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),

te izlazne signale:

- *random_num[23..0]* - generisani slučajni broj (24 bita).



Slika 28. Komponenta RNG

SMAC Controller

Komponenta SMAC Controller predstavlja upravljačku jedinicu S-MAC protokola. Princip rada ove komponente je opisan pomoću SDL dijagrama u poglavlju "S-MAC protokol". Na slici 29 je dat šematski prikaz komponente SMAC Controller. Komponenta posjeduje sljedeće ulazne signale:

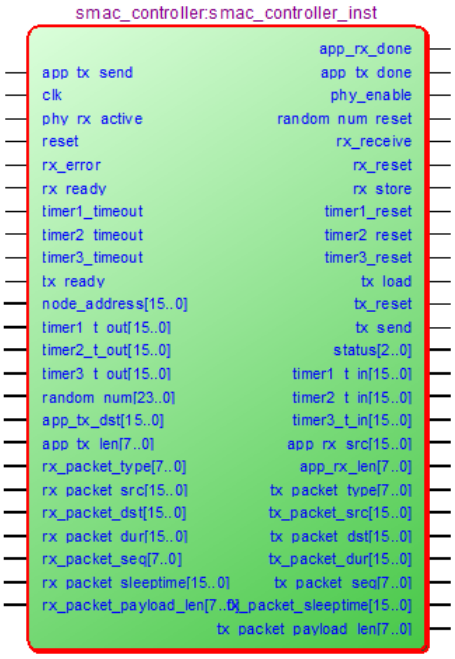
- *app_tx_send* - signal za početak procedure slanja DATA okvira (aktivan u stanju logičke "1"),
- *clk* - signal takta,
- *phy_rx_active* - signal koji potvrđuje da je u toku prijem podataka u UART Controller-u (aktivan u stanju logičke "1"),
- *reset* - asinhroni reset signal (aktivan u stanju logičke "1"),
- *rx_error* - signal koji indicira da je došlo do greške u prijemu okvira u SMAC Rx Controller-u (aktivan u stanju logičke "1"),
- *rx_ready* - signal koji potvrđuje da je SMAC Rx Controller spreman za prijem okvira (aktivan u stanju logičke "1"),
- *timer1_timeout* - signal koji indicira da je istekao brojač 1 (aktivan u stanju logičke "1"),

- *timer2_timeout* - signal koji indicira da je istekao brojač 2 (aktivan u stanju logičke "1"),
- *timer3_timeout* - signal koji indicira da je istekao brojač 3 (aktivan u stanju logičke "1"),
- *tx_ready* - signal koji potvrđuje da je SMAC Tx Controller spreman za slanje okvira (aktivan u stanju logičke "1"),
- *node_address[15..0]* - adresa bežičnog senzorskog čvora (16 bita = 2 okteta, podešava se na najvišem nivou hijerarhije),
- *timer1_t_out[15..0]* - preostalo vrijeme brojača 1 (16 bita = 2 okteta),
- *timer2_t_out[15..0]* - preostalo vrijeme brojača 2 (16 bita = 2 okteta),
- *timer3_t_out[15..0]* - preostalo vrijeme brojača 3 (16 bita = 2 okteta),
- *random_num[23..0]* - slučajni broj koji je generisao RNG (24 bita),
- *app_tx_dst[15..0]* - odredišna adresa DATA okvira koji se šalje (16 bita = 2 okteta),
- *app_tx_len[7..0]* - dužina aplikacijskog dijela DATA okvira koji se šalje (8 bita => maks. dužina 255 uključujući zaglavlje i CRC polje DATA okvira),
- *rx_packet_type[7..0]* - tip paketa koji je primljen (8 bita = 1 oktet),
- *rx_packet_src[15..0]* - izvorišna adresa paketa koji je primljen (16 bita = 2 okteta),
- *rx_packet_dst[15..0]* - odredišna adresa paketa koji je primljen (16 bita = 2 okteta),
- *rx_packet_dur[15..0]* - trajanje transmisije paketa (16 bita = 2 okteta),
- *rx_packet_seq[7..0]* - redni broj paketa koji je primljen (8 bita = 1 oktet),
- *rx_packet_sleeptime[15..0]* - vrijeme spavanja čvora koji se poslao paket (16 bita = 2 okteta),
- *rx_packet_payload_len[7..0]* - dužina aplikacijskog dijela paketa koji je primljen (8 bita = 1 oktet),

te izlazne signale:

- *app_rx_done* - signalizira da je primljen DATA okvir (aktivan u stanju logičke "1"),
- *app_tx_done* - signalizira da je uspješno poslan i isporučen DATA okvir (aktivan u stanju logičke "1"),
- *phy_enable* - signal za omogućavanje rada RF primopredajnika (aktivan u stanju logičke "1"),
- *random_num_reset* - signal za reset RNG-a (aktivan u stanju logičke "1"),
- *rx_receive* - signal za aktiviranje prijemnika okvira (aktivan u stanju logičke "1"),
- *rx_reset* - signal za reset SMAC Rx Controller-a (aktivan u stanju logičke "1"),
- *rx_store* - signal za spremanje aplikacijskog dijela DATA okvira iz SMAC Rx Controller-a (aktivan u stanju logičke "1"),
- *timer1_reset* - signal za reset brojača 1 (aktivan u stanju logičke "1"),
- *timer2_reset* - signal za reset brojača 2 (aktivan u stanju logičke "1"),
- *timer3_reset* - signal za reset brojača 3 (aktivan u stanju logičke "1"),
- *tx_load* - signal za učitavanje aplikacijskog dijela DATA okvira u SMAC Tx Controller (aktivan u stanju logičke "1"),
- *tx_reset* - signal za reset SMAC Tx Controller-a (aktivan u stanju logičke "1"),
- *tx_send* - signal za početak slanja okvira iz SMAC Tx Controller-a (aktivan u stanju logičke "1"),
- *status[2..0]* - signalizira stanje SMAC Controller-a (mapirano na 3 LED-ice na razvojnom sistemu),

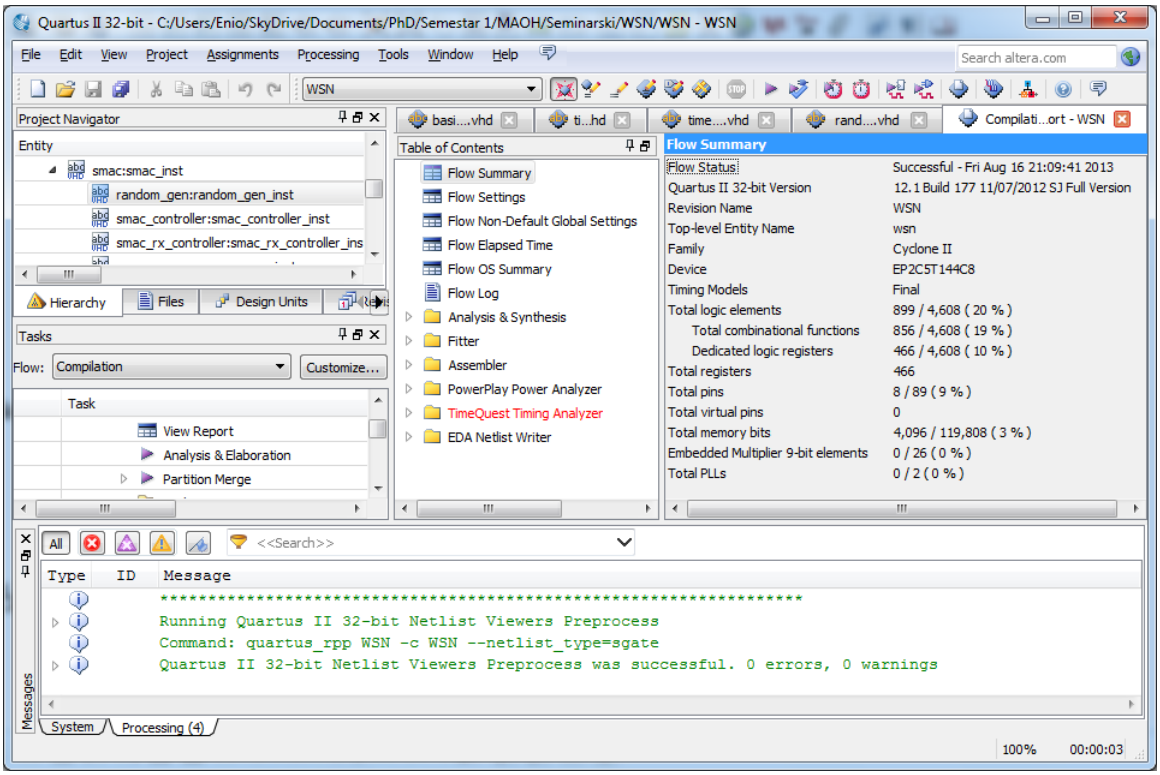
- *timer1_t_in[15..0]* - postavljanje vrijednosti brojača 1 (16 bita = 2 okteta),
- *timer2_t_in[15..0]* - postavljanje vrijednosti brojača 2 (16 bita = 2 okteta),
- *timer3_t_in[15..0]* - postavljanje vrijednosti brojača 3 (16 bita = 2 okteta),
- *app_rx_src[15..0]* - izvorišna adresa primljenog DATA okvira (16 bita = 2 okteta),
- *app_rx_len[7..0]* - dužina aplikacijskog dijela primljenog DATA okvira (8 bita => maks. dužina 255 uključujući zaglavlje i CRC polje DATA okvira),
- *tx_packet_type[7..0]* - tip paketa koji se šalje (8 bita = 1 oktet),
- *tx_packet_src[15..0]* - izvorišna adresa paketa koji se šalje (16 bita = 2 okteta),
- *tx_packet_dst[15..0]* - odredišna adresa paketa koji se šalje (16 bita = 2 okteta),
- *tx_packet_dur[15..0]* - trajanje transmisije paketa (16 bita = 2 okteta),
- *tx_packet_seq[7..0]* - redni broj paketa koji se šalje (8 bita = 1 oktet),
- *tx_packet_sleeptime[15..0]* - vrijeme spavanja čvora koji šalje paket (16 bita = 2 okteta),
- *tx_packet_payload_len[7..0]* - dužina aplikacijskog dijela paketa koji se šalje (8 bita = 1 oktet).



Slika 29. Komponenta SMAC Controller

Pregled iskorištenih resursa na FPGA čipu

Za sintezu VHDL koda korišten je Altera Quartus II 12.1. Na slici xx prikazan je sažetak izvještaja o sintezi u kojem se može očitati iskorištenost resursa na FPGA čipu. Kompletna implementacija S-MAC protokola je iskoristila 899 logičkih elemenata.



Slika 29. Sažetak izvještaja o sintezi VHDL koda

Zaključak

Cilj ovog rada, implementacija protokola za kontrolu pristupa mediju u bežičnim senzorskim mrežama na FPGA, je postignut na sljedeći način:

- dat je pregled najvažnijih zahtjeva za dizajn protokola za kontrolu pristupa mediju u bežičnim senzorskim mrežama,
- uvažavajući date zahtjeve odabran je S-MAC protokol,
- dizajn S-MAC protokola je izložen pomoću vremenskih i SDL dijagrama koji pokrivaju sve moguće slučajeve upotrebe,
- predložena je arhitektura sistema kao osnova za implementaciju S-MAC protokola na FPGA,
- odabrana je hardverska platforma za implementaciju i testiranje protokola,
- komponente sistema su realizirane pomoću VHDL jezika.

Iz izvještaja o sintezi VHDL koda očitano je da je kompletna implementacija S-MAC protokola iskoristila 899 logičkih elemenata. Najjednostavnija mikroprocesorska arhitektura Altera Nios II/e se sastoji iz sljedećih komponente [9]:

- mikroprocesor Nios II/e (540 logičkih elemenata),
- UART kontroler (140 logičkih elemenata),
- RAM kontroler (750 logičkih elemenata),
- Timer (150 logičkih elemenata),

što zahtijeva ukupno 1580 logičkih elemenata. Poređenjem iskorištenosti resursa za ova dva pristupa može se zaključiti da je postavljena hipoteza koja glasi "Protokol za kontrolu pristupa mediju je moguće realizirati kao hardversku strukturu koja će upotrijebiti manje logičkih elemenata FPGA čipa nego što bi bilo potrebno za realizaciju najjednostavnije Altera Nios mikroprocesorske arhitekture." potvrđena.

Literatura

[1] Robert Abo, Kamel Barkoui, and Hafedh Zayani, *A probabilistic analysis of energy efficiency of wireless sensor network duty-cycled MAC protocols: application to S-MAC and Eco-MAC*, In Proceedings of the 6th ACM workshop on QoS and security for wireless and mobile networks (Q2SWinet '10), ACM, New York, NY, USA, 2010.
[2] Koubaa, A., Alves, M., Tovar , E., *Lower Protocol Layers for Wireless Sensor Networks: A Survey*, IPPHURRAY Technical Report, HURRAY-TR-051101, 2005.
[3] Simarpreet Kaur and Leena Mahajan, *Power Saving MAC Protocols for WSNs and Optimization of S-MAC Protocol*, International Journal of Engineering Business Management, W.H. Ip (Ed.), ISBN: 1847-9790, InTech, 2012.
[4] Rajgopal Kannan, Ram Kalidindi, S. S. Iyengar, and Vijay Kumar, *Energy and rate based MAC protocol for wireless sensor networks*, SIGMOD Rec. 32, 4, 2003.
[5] Joseph Kabara and Maria Calle, *MAC Protocols Used by Wireless Sensor Networks and a General Method of Performance Evaluation*, International Journal of Distributed Sensor Networks, vol. 2012, Article ID 834784, 11 pages, 2012.
[6] M. Riduan Ahmad, Eryk Dutkiewicz and Xiaojing Huang, *A Survey of Low Duty Cycle MAC Protocols in Wireless Sensor Networks*, Emerging Communications for Wireless Sensor Networks, Anna Foerster and Alexander Foerster (Ed.), ISBN: 978-953-307-082-7, InTech, 2011.
[7] Altera Corporation, *Cyclone II Device Handbook, Volume 1*, San Jose, CA, USA, 2008.
[8] Web stranica, *http://www.electronicdesignworks.com/utilities/crc_generator_doc/crc_generator_doc.htm*, pristup ostvaren 22.09.2013.
[9] Altera Corporation, *Nios II Performance Benchmarks*, San Jose, CA, USA, 2013.