Meteorološka mjerna postaja

Kantoci, Kruno

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

DIPLOMSKI RAD

Kruno Kantoci

Zagreb, 2009.

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

DIPLOMSKI RAD

Voditelj rada:

Prof. dr. sc. Mladen Crneković

Kruno Kantoci

Zagreb, 2009.

Sažetak

Ovaj rad prati i opisuje razvoj prototipne meteorološke postaje, koja daje trenutne informacije o vrijednostima tlaka, temperature, relativne vlažnosti te brzini i smjeru vjetra. Da bi se lakše razumjela njihova međuovisnost, na samom početku rada opisana je struktura atmosfere te parametri koji utječu na promjenu njenog stanja. Također, dana je i cijela slika razvoja meteorologije kao znanosti kroz povijest, sa velikim naglaskom na suvremene metode mjerenja, obrade i analize dobivenih podataka.

Elektronički i mehanički gledano, postaja se sastoji od dvije cjeline: vanjske jedinice, koja u sebi ima implementirana mjerna osjetila i sklopove za prilagodbu signala, te unutarnje jedinice, koja obrađuje dobivene podatke te ih ispisuje na LCD ekranu i šalje preko serijske veze prema računalu. Svi korišteni elektronički elementi, kao što su senzori, sklopovi za prilagodbu signala, LCD ekran te konačno mikrokontroler, detaljno su opisani. Kako je teško sagledati cijeli uređaj kao skup zasebnih dijelova, velika pozornost je usmjerena i na njihovu međusobnu interakciju.

Izrada elektroničke platforme za sve senzore te sklopove za prilagodbu signala izvedena je pomoću računalne aplikacije Protel. Modelirana elektronička pločica za pojedini element prikazana je prilikom njegovog opisivanja, da bi se što lakše i brže mogla uvidjeti veza između zahtjeva prije modeliranja i konačne modelirane pločice. Sve su pločice izrađene foto postupkom na standardnim Euro pločicama 100x160 mm s foto premazom.

Unutarnja jedinica služi za obradu svih dobivenih podataka, što se izvodi pomoću mikrokontrolera tvrtke Atmel, tipa AT89C51ID2. Njegovo programiranje izvedeno je u programskom paketu MicroVision koji koristi kodnu sintaksu C – a. Da bi se podaci primljeni računalom mogli svrsishodno prikazati, razvijena je računalna aplikacija „Meteo“, koja u analogno/digitalnom obliku prikazuje trenutne izmjerene vrijednosti, te također iste sprema kao tekstualne datoteke da bi se naknadno mogli pogledati i, ako je potrebno, grafički prikazati. Za njen razvoj korišten je programski paket Delphi7, koji je usmjeren ka objektnom programiranju, pri čemu se koristi kodna sintaksa Pascala. Svi koraci pri programiranju detaljno su opisani, a u prilogu se nalaze cijeli programski kodovi za mikrokontroler i razvijenu računalnu aplikaciju.

Svi mehanički i elektronički elementi modelirani su u računalnom programu Catia, čime je prikazan raspored elemenata unutar mjerne meteorološke postaje.

Na posljetku, prikazane su cijene pojedinih stavki korištenih pri izradi ovog rada, te konačna cijena razvijenog prototipa meteorološke mjerne postaje.

Sadržaj

Popis slika

Popis tablica

Izjava

Izjavljujem da sam diplomski rad na temu „Meteorološka mjerna postaja“ izradio samostalno, koristeći navedenu stručnu literaturu i znanje stečeno tijekom dosadašnjeg školovanja.

Zahvala

Zahvaljujem prof. dr. sc. Mladenu Crnekoviću na uloženom trudu i vremenu, ne samo za vrijeme izrade diplomskog rada, već od samog početka studija. Također velika zahvala ide gospodinu Zvonku Grgecu koji mi je praktičnim savjetima olakšao izradu ovog rada.

Neizostavni tokom mog školovanja su moji roditelji i brat, koji su mi pružili financijsku i moralnu pomoć, te se radovali svakom mom uspjehu. Bez njih ovaj rad ne bi bio pisan.

Velika zahvala i mojoj djevojci koja je sve vrijeme bila uz mene kao moralna potpora, te svim prijateljima koji su bili uz mene.

Svima od srca zahvaljujem!

1 UVOD

Jedna od prvih informacija koja nam je potrebna kada se ujutro probudimo je informacija o vremenu. Davno su prošla vremena „kamen prognoze“ ili „prst prognoze“. To možemo zahvaliti svjetskoj mreži meteoroloških postaja. Današnji oblici informiranja kao što su internet, radio ili TV svaki sat daju informaciju o trenutnom vremenu. Dakako, te informacije isključivo ovise o lokaciji meteorološke postaje tako da se podaci mogu dosta razlikovati od onih u našoj sredini. Kako je, npr. tlak jedan od parametara koji se jako razlikuje s porastom visine, meteorološko pravilo je da se mora svesti na visinu razine mora.

Da bi se dobila točna informacija o meteorološkim parametrima u našoj okolini, razvijen je prototip meteorološke mjerne postaje. Ona zamjenjuje potrebu paljenja računala ili nekog drugog uređaja, a pri tome nam daje točnu informaciju o trenutnom stanju atmosfere gdje točno i kada točno mi to želimo. Kako je preko veze s računalom omogućeno spremanje svih dobivenih parametara, moguće je vidjeti kretanje atmosferskih parametara te, ako je potrebno, jednostavnom aplikacijom poslati ih preko interneta na drugi kraj svijeta. Također, ovaj uređaj nudi mogućnost spajanja sa nekim drugim uređajima, čijom se kombinacijom može upravljati sustavom za postizanje optimalnih uvjeta u kući ili spriječiti uništavanje prozora pravovremenim spuštanjem roleta.

Dakako, slične mjerne meteorološke postaje već postoje na tržištu. U donjoj tablici (Tablica 1) navede su meteorološke mjerne stanice najpoznatijih svjetskih proizvođača, njihove karakteristike i cijena.

Tablica 1. Meteorološke postaje drugih proizvođača

Proizvođač	**RainWise**	**Davis Instruments**	**Oregon Scientific**
Model	MK-III-RTN	Vantage Pro2	WMR 200
Slika			
Temperatura [°C]	- 54 do + 74	- 40 do + 65	- 30 do + 60
Relativna vlažnost [%]	0 do 100	0 do 100	0 do 100

Tlak [hPa]	551 do 1084	880 do 1080	700 do 1050
Smjer vjetra	✓	✓	✓
Brzina vjetra [m/s]	0 do 67	1 do 67	3 do 56
Ostale značajke	• bežično povezivanje • sat i kalendar • unutarnja temperatura • trend tlaka • spremanje podataka	• bežično povezivanje • sat i kalendar • unutarnja temperatura • trend tlaka • sunčevo zračenje • mjesečeve mijene • količina oborine • izlazak/zalazak sunca • spremanje podataka • UV indeks	• bežično povezivanje • sat i kalendar • unutarnja temperatura i vlažnost • sunčevo zračenje • mjesečeve mijene • količina oborine • UV indeks • prognoza vremena za sljedećih 12 h • spremanje podataka
Web stranica proizvođača	www.rainwise.com	www.davisnet.com	www.oregonscientific.com
Cijena (okvirno kn)	5000	3800	2300

2 METEOROLOGIJA

Meteorologija (vremenoslovlje) je znanost o Zemljinoj atmosferi i promjenama u njoj. Meteorologija proučava promjene vremena oko nas. Pripada u skupinu geofizičkih znanosti. Neke od glavnih pojava koje se proučavaju su količina i vrsta oborina, grmljavinske oluje, tornada, tropski cikloni i tajfuni. Bitan utjecaj vremena na ljude i ljudske aktivnosti doveo je do razvoja znanosti o prognoziranju vremena.

2.1 Povijesni razvoj meteorologije

Riječ meteorologija potječe od grčke riječi *meteoron* koja se odnosila na sve pojave na nebu. Zanimanje čovjeka za vrijeme koje ga okružuje postojalo je otkad i sam čovjek. Već u staroj Kini, Indiji, Egiptu i Grčkoj su ljudi raspravljali o vjetrovima i oborinama te pokušavali shvatiti i objasniti te vremenske pojave. Prva knjiga s opisom i tumačenjem vremenskih pojava je Aristotelova *Meteorologica* (340. g. prije Krista), a obuhvaćala je sve pojave iznad tla. Idućih stoljeća, skoro cijelo tisućljeće, meteorologija se nije uopće ili se vrlo slabo razvijala. Iz tog vremena postoje rijetki zapisi, uglavnom crkveni, o vremenskim pojavama i posebno nepogodama.

Počeci meteorologije leže u promatranju trenutnog vremena i nagađanja kakvo bi ono moglo biti u vrlo bliskoj budućnosti. Aristotelov nauk i njegova *Meteorologica* bili su u antici i srednjem vijeku vrlo cijenjeni i zapravo jedini koliko-toliko znanstveni meteorološki počeci. Tako je bilo sve dok René Descartes, Galileo Galilej i ostali nisu nagađanja počeli zamjenjivati instrumentalnim promatranjima početkom 17. stoljeća. Najosnovniji instrumenti za provođenje tih promatranja i mjerenja, barometar, higrometar i termometar, izumljeni su u razdoblju između 1650. i 1750. godine. Spajanje teorije i eksperimenta uključivalo je i Newtonove zakone gibanja, pokuse Blaisa Pascala, Edmea Marriottea, Roberta Hookea, Edmunda Halleya i ostalih, zatim istraživanja Roberta Boylea s plinovima te Halleya, Georga Hadleya i Jeana Le Rond d'Alemberta o atmosferskoj cirkulaciji.

Tijekom sljedećeg stoljeća (1750.-1850.) standardizirani su termometri, Benjamin Franklin proučavao je munje i izumio gromobran, John Dalton postavio je temelje za mjerenje isparavanja i vlažnosti, a Luke Howard je klasificirao oblake. Nakon 1800. godine javne ustanove, ali i fizičke osobe počele su skupljati i pratiti vremenske prilike.

Nakon što je u Krimskom ratu (1853.-1856.) francuska flota bila teško oštećena u snažnoj oluji, zemlje zapadne Europe i Sjeverne Amerike započele su ozbiljne pokušaje skupljanja podataka o vremenu na mnogo mjesta istovremeno pomoću nedavno izumljenog

telegrafa (1837.). Razvoj pouzdanih satova omogućio je stalnost i točnost promatranja na širem području. Izumljeni su i anemometri, a uskoro je za održanje i očitavanje uređaja uvedena i električna struja. S razvojem prometa baloni, zmajevi i zrakoplovi uskoro su na svojim letovima nosili i meteorološke instrumente kroz troposferu, najniži sloj Zemljine atmosfere, sve do stratosfere, idućeg sloja atmosfere. Stratosfera je otkrivena, opisana i nazvana malo nakon 1900. godine. Stalna mjerenja po visini započela su oko 1920. godine, nakon što su izumljeni radio uređaji na baterije koji su bili postavljani na balone. Podaci o stanju vremena na većim visinama dali su potpuniju sliku stanja atmosfere i bolji uvid u pojave na tim visinama, poput mlazne struje.

Termodinamika, koja se počela razvijati sredinom 19. stoljeća, omogućila je velik broj novih formula koje opisuju atmosferu i promjene u njoj. Od 1850. do 1950. godine dominantna grana meteorologije bila je sinoptička meteorologija. Oko 1920. empirijska iskustva prepuštaju mjesto fizici, a znanstvenici Vilhelm Bjerkness i njegov sin Jacob sve te ideje oblikovali su u teoriju o polarnoj fronti, uključujući ključne pojmove fronte i zračnih masa.

Moderna dinamička meteorologija rođena je 1948. godine, kad je Jule Charney uspio reducirati složene dinamičke jednadžbe (koje je već 1904. godine postavio Vilhelm Bjerkness) na jednostavniji, ali korisni oblik. Istovremeni razvoj digitalnog računala osigurao je da Charneyeva metoda rješavanja jednadžbi ima veliku praktičnu korisnost jer se omogućilo da prognoziranje vremena bude bazirano na rješenjima dinamičkih jednadžbi kao funkcija vremena. Od 1948. naglo se razvija i radarska tehnologija pa se već 1950. godine radarima moglo razlikovati sastav oblaka po količini vode u njima i tako detektirati oluje, osobito one grmljavinske. Od sredine šezdesetih godina izumljeni su i radari koji su Dopplerovim efektom davali informacije i o brzini. Nakon 1960. sateliti su počeli slati detaljne slike cijele Zemljine površine.

2.2 Današnja meteorologija

S razvitkom meteorologije otvorila se i mogućnost njenog iskorištavanja u svakodnevnom životu za potrebe čovjeka. To je potaknulo organizaciju i nastanak prvih meteoroloških službi, ali i razvilo spoznaju o velikoj važnosti međunarodne suradnje. Ljudi su brzo shvatili da vrijeme i meteorološka zbivanja ne poznaju državne granice i da prelaze granice kontinenata. Međunarodna povezanost u meteorologiji utemeljena je na 1. međunarodnom kongresu meteorologa u Beču 1873. godine, gdje je osnovana Međunarodna

meteorološka organizacija (International Meteorological Organization – IMO). Ta organizacija je 1951. godine prerasla u Svjetsku meteorološku organizaciju (World Meteorological Organization – WMO), posebnu agenciju Ujedinjenih naroda. 1. rujna 1993. WMO je obuhvaćao 167 država i 5 teritorija članica, uključujući i Hrvatsku. Zadaća je Svjetske meteorološke organizacije (WMO) sudjelovati u organiziranju mreže meteoroloških postaja na kojima će se mjeriti i opažati meteorološki elementi i pojave na jedinstven način, sudjelovati u organiziranju sustava brze razmjene meteoroloških izvješća, organizirati znanstvena istraživanja te pomagati primjenu meteorologije u svim ljudskim djelatnostima.

Meteorologija kao znanost i dalje se razvija. Od velike pomoći je i nagla kompjuterizacija i automatizacija, pogotovo u iskorištavanju ogromnog broja motrenja koja se dnevno obavljaju tradicionalnim, ali i novim instrumentima. Npr. razvoj Doppler radara ključan je za pravodobna i što točnija upozorenja za nadolazeći tornado ili druge lokalne vremenske događaje koji predstavljaju opasnost ljudima i imovini. Nova supermoćna računala jedina mogu u vrlo kratkom vremenu procesuirati mnoštvo informacija koje svakog trenutka stižu sa svih strana svijeta, a to je ključno za pravovremeno i točno rješavanje složenih jednadžbi koje opisuju i predviđaju stanje atmosfere. Određen broj svih tih informacija širi se svijetom posredstvom Globalnog telekomunikacijskog sustava Svjetske meteorološke organizacije (WMO), ali dobar dio se ne šalje u javnost zbog komercijalnog interesa, nacionalne sigurnosti i logistike nekih zemalja. Iz tog razloga diljem svijeta postoji nekoliko središta koja pomoću brzih i moćnih računala i računalnih modela izvode simulacije vremena u budućnosti temeljene na dosadašnjim opažanjima. Jedno od tih središta je i Europski centar za srednjoročnu prognozu vremena (ECMWF) u Bracknellu u Engleskoj.

Vrlo bitan dio meteorologije su meteorološka opažanja i mjerenja. Ona se vrše na mnoge načine, najčešće u meteorološkim postajama, a od velike važnosti u novije vrijeme su radio, radar i umjetni sateliti. Računalna tehnologija uspješno se i uvelike koristi, uključujući numeričke modele, interaktivnu analizu podataka i njihovo potpuno razumijevanje. Meteorologija djeluje u vezi s mnogim granama znanosti koje se bave čovjekovom okolinom. Neke od važnijih su aeronautika, agrikultura, arhitektura, ekologija, proizvodnja energije, šumarstvo, hidrologija, medicina i oceanografija. Mnoge od navedenih znanosti uvelike ovise o učincima vremena na određenom mjestu, no hidrologija i oceanografija npr. utječu i povratno na meteorologiju jer svojim učincima mijenjaju i atmosferske uvjete na Zemljinoj površini.

2.3 Meteorološka motrenja

Da bi se što bolje upoznalo ponašanje i predvidio razvoj atmosferskih procesa, meteorološki se elementi neprekidno mjere na cijeloj Zemlji. Mjerenja se većinom svrstavaju u prizemna, visinska i daljinska, iako te skupine nisu strogo razdvojene. Obavljaju se na meteorološkim postajama posebno odabranih položaja. Kriteriji za gustoću postaja i reprezentativnosti njihova položaja utvrđeni su na međunarodnoj razini i ovise o vrsti i namjeni postaje. Visinske postaje jedna od druge mogu biti udaljene do 300 km, a prizemne su na deseterostruko manjoj udaljenosti. Razvoj meteoroloških motrenja za budućnost ide u tom smjeru da se vizualna opažanja po mogućnosti zamijene instrumentalnim mjerenjima, a mjerenja da se što više automatiziraju. Automatske meteorološke postaje i meteorološki sateliti zajedno s prijenosnim uređajima postaju dio informacijskog sustava svake meteorološke službe.

Prizemna motrenja sastoje se od mjerenja instrumentima unutar sloja zraka do 10 m visine i opažanja u vidokrugu motritelja. Jedan od kriterija za određivanje gustoće mreže prizemnih meteoroloških postaja jest da cijeli nebeski svod i Zemljina površina budu pokriveni opažanjima i mjerenjima susjednih postaja kako bi se dobila neprekinuta slika o poljima meteoroloških elemenata na većem području. Postaja mora biti reprezentativna, što znači da na podatke ne utječu usko lokalne okolnosti, npr. drveće, građevine i slično. U Hrvatskoj ima tridesetak sinoptičkih i aerodromskih postaja, više od sto klimatoloških i oko četiri stotine kišomjernih. Uži prostor za postavljanje instrumenata i motrenje treba biti u prirodnom okolišu. Mjeri se starijim klasičnim instrumentima i modernijima, automatiziranim mjernim sustavima.

2.3.1 Mjerenje atmosferskog tlaka

Atmosferski tlak mjeri se barometrima i iskazuje u hektopaskalima. Barometar se postavlja u prostoriju bez velikih kolebanja temperature i jačeg strujanja zraka. Na živinom se barometru vizirom s noniusom određuje visina stupca žive (u ravnoteži sa stupcem zraka) na desetinku jedinica skale. Većinom su na skali milimetri, pa je potrebna konverzija u hektopaskale. Tlak je definiran visinom stupca žive pri 0 °C, pa se mora očitati i temperatura barometra na pridodanom termometru i s pomoću unaprijed izračunatih tablica ili računalnog programa, svesti izmjereni podatak na 0 °C. Kako tlak ovisi o nadmorskoj visini, ne mogu se uspoređivati mjerenja na raznim postajama prije nego što su tlakovi svedeni na istu razinu. Za

postaje do 700 m visine to je morska razina, a za više postaje najbliža standardna izobarna ploha. Ulazni podaci za to svođenje su tlak, temperatura i vlažnost zraka te visina barometra.

U aneroid – barometrima osjetilo je elastična kutija iz koje je djelomično isisan zrak. Takva osjetila upotrebljavaju se uglavnom u automatskim električnim mjernim uređajima i u instrumentu za neprekidno bilježenje atmosferskog tlaka – barografu. Takvi barometri su temperaturno kompenzirani, pa očitane vrijednosti ne treba svoditi na 0 °C.

2.3.2 Mjerenje temperature zraka

Temperatura zraka mjeri se živinim, katkad alkoholnim, termometrima na desetinku stupnja točno, 2 m iznad tla u termometarskoj kućici, gdje je osigurano slobodno prirodno strujanje zraka i zaštita od sunca. Maksimalna temperatura mjeri se gotovo vodoravno postavljenim maksimum – termometrom sa suženom kapilarom na izlazu iz rezervoara. Tu se pri snižavanju temperature živina nit prekida pa zaostala živa u kapilari pokazuje kolika je bila najviša dostignuta vrijednost temperature. Minimum – termometar se postavlja vodoravno, a punjen je alkoholom u kojemu je štapić posebnog oblika. Kad temperatura pada, rub alkohola u kapilari povlači sa sobom štapić. Kad temperatura raste, alkohol teče mimo štapića i on ostaje na mjestu koje označuje najnižu temperaturu.

2.3.3 Mjerenje vlažnosti zraka

Za istodobno mjerenje temperature zraka i određivanje parametara vlažnosti zraka služi psihometar. Sastoji se od suhog termometra, koji pokazuje temperaturu zraka, i mokroga, koji na rezervoaru ima krpicu vlaženu vodom, pa zbog isparavanja vode s krpice, ovisno o količini vodene pare u zraku, pokazuje nižu temperaturu. Obično je aspiriran radi osiguranja jednoličnog strujanja zraka preko termometra. Iz temperature suhog i mokrog termometra određuje se tlak vodene pare, relativna vlažnost i rosište pomoću tablica ili računala.

2.3.4 Mjerenje vjetra

Vjetar se u principu mjeri na visini 10 m nad otvorenim terenom. Vjetrulja pokazuje odakle puše vjetar. Puni se krug dijeli na 16, 32 ili 356 smjerova. Brzina vjetra mjeri se najčešće anemometrom sa šalicama koje se okreću, a katkad s propelerom. Brzini vrtnje razmjeran je proizvedeni električni napon, odnosno broj električnih ili mehaničkih kontakata u jedinici vremena. Od anemometra i vjetrulje bolji su mehanički, električni i elektronski

anemografi koji bilježe smjer i brzinu vjetra. Za posebna mjerenja upotrebljavaju se i ručni anemometri u kojima vjetar hladi električno grijanu žicu pa se mjeri otpor ili se mjeri razlika dinamičkog tlaka što ga proizvodi vjetar prema statičkom tlaku.

3 ATMOSFERA

Atmosfera je smjesa plinova koja obavija Zemlju. Premda i na visini od 1000 km još ima poneka molekula atmosferskih plinova, 99,9% mase cijele atmosfere zbijeno je, zbog sile teže, u donjih 50 km. Atmosfera je najgušća tik uz Zemljinu površinu. Vrijeme i klimu oblikuju fizička svojstva, procesi i pojave u najnižih deset kilometara. Dio atmosfere, osobito u malim visinama, nazivamo zrakom. Gustoća zraka ovisi o njegovoj temperaturi, tlaku i vlažnosti, pa suhi zrak pri 0 °C i tlaku 1013,25 hPa ima gustoću 1,292 kg/m^3. U tom je slučaju obujam 1 kg zraka 774 m^3. Vlažni zrak ima manju gustoću od suhoga.

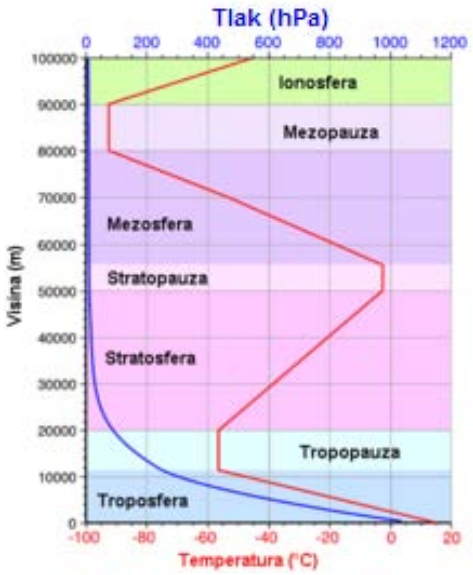
3.1 Sastav atmosfere

U donjih osamdesetak kilometara visine atmosferski su plinovi jednoliko izmiješani. To znači da se u bilo kojoj visini tog dijela atmosfere obujemni udjeli većine plinova uvijek jednako odnose. Udjel triju plinova – dušika (78%), kisika (21%) i argona (0,9%) iznosi 99,9%, a udjeli ugljik – dioksida i ozona su promjenjivi. U svojim donjim slojevima atmosfera nikad nije posve suha, već sadrži više ili manje vodene pare. Obujmeni udjel vodene pare može biti i do 4%.

U atmosferi ima mikroskopski sitnih čvrstih i tekućih čestica (aerosola, lebdećih čestica) koje lebde i nošene su zračnim strujama. Prirodni aerosol djelomično dolazi iz svemira kao svemirska prašina, a većim dijelom potječe sa Zemljine površine kao prašina s tla, kristalići morske soli, čestice iz vulkanskih erupcija, pelud, spore, bakterije, virusi i sl. Toj skupini pripadaju i kapljice vode te kristalići leda, koji u većoj koncentraciji čine maglu i oblake. Važan i po pravilu štetan dio aerosola rezultat je ljudskih djelatnosti (prometa, tvorničke proizvodnje, gradnje i sl.). Ljudskom djelatnošću u atmosferu ulaze i neki plinovi štetni za zdravlje i život na Zemlji.

3.2 Podjela atmosfere po visini

Atmosfera se prema temperaturi dijeli na četiri glavna i tri prijelazna sloja (Slika 1).



Slika 1. Promjena temperature i tlaka s visinom u atmosferi

Troposfera je najniži sloj. Debljina joj ni prostorno ni vremenski nije stalna, a gornja joj se granica od ekvatora do polova spušta s približno 17 km na nekih 9 km visine. U troposferi se zbiva ono što nazivamo vremenom. Niži dijelovi troposfere pod velikim su utjecajem Zemljine površine. Iz njih u zrak ulazi vodena para, aerosol i ostale primjese. Neravnine na tlu prepreke su zračnim strujama, te ih tako mijenjaju. Najvažnije je da je tlo danju izvor, a noću ponor topline tako da djeluje na temperaturu zraka koja se većinom, ali ne uvijek, smanjuje s porastom visine. Vertikalnu promjenu temperature kad hladniji zrak leži ispod toplijeg zovemo *temperaturnom inverzijom*. U povoljnim uvjetima, a to su duga noć, slab vjetar, relativno suh zrak, snijeg na tlu i udubljeni teren, pri tlu se taloži nekoliko stotina metara debeo sloj hladnog zraka. Nejednoliko grijanje Zemljine površine osnovni je uzrok gibanju zraka u troposferi. Zračne čestice u pokretu prenose toplinu, vlagu i razne primjese. Pritom voda može mijenjati agregatno stanje, a s tim u vezi nastaju ne samo oblaci različitog izgleda i oborina, nego i raznolike optičke i električne pojave (duga, halo, munje...).

Zbog posebnih, karakterističnih svojstava unutar troposfere razlikujemo slojeve koji nisu strogo omeđeni niti su im debljine stalne. To je ponajprije *prizemni sloj*, ispod 2 m visine, u kojemu mikroklima izrazito ovisi o najbližim lokalnim okolnostima – sastavu i obliku tla. Zato se standardna meteorološka mjerenja obavljaju tik iznad tog sloja. U prizemnom su sloju promjene temperature između dana i noći velike, a temperaturna je inverzija noću gotovo redovita. Meteorološke okolnosti u prizemnom sloju osobito snažno utječu na biljni svijet. *Planetarni granični sloj* mnogo je deblji i prostire se od tla do

prosječno 1 km visine. Odlikuje se vrtloženjem zraka zbog utjecaja podloge. Riječ je o različito orijentiranim vrtlozima promjera od nekoliko centimetara do više stotina metara, tako da se zrak komeša u svim smjerovima. Vjetar, te neravna i ugrijana podloga pogoduju vrtloženju zraka. Što je ono jače, to se dalje rasprostiru primjese dospjele u atmosferu, a njihova se koncentracija u blizini izvora smanjuje. Planetarni granični sloj noću je debeo samo nekoliko stotina metara, a danju može doprijeti i do 2,5 km iznad tla. Iznad njega je *slobodna troposfera*, u kojoj je utjecaj trenja o podlogu zanemariv i zato je brzina vjetra veća. U tom se sloju temperatura svakih 100 metara visine smanjuje za 0,5 ili 0,6 °C. Na gornjoj granici troposfere iznad polova iznosi oko -45 °C, a iznad ekvatora oko -80 °C.

Stratosfera je iznad troposfere i seže do približno 50 km visine, gdje je temperatura opet slična onoj pri tlu. U stratosferi gotovo nema vertikalnog miješanja zraka i vjetar je pretežno zapadni. Ako u donju stratosferu ipak uđe neko onečišćenje, ono će se u tom sloju zadržati mjesecima, pa i godinama. To se prirodno događa pri jakim provalama vulkana a čovjekovim utjecajem u taj sloj dopiru freoni koji razaraju za život vrlo važan sloj bogat ozonom. Povremeno i zrak iz stratosfere može doprijeti u troposferu.

U **mezosferi**, sloju od približno 50 do otprilike 80 km visoko iznad tla, temperatura se naglo smanjuje. Tako je gornja granica mezosfere, s temperaturom otprilike -90 °C, najhladniji dio atmosfere. Iznad nje se nalazi **termosfera**, u kojoj je zrak vrlo rijedak, a dnevni raspon temperature izrazito velik.

Na ulazu u atmosferu Sunčevo zračenje ionizira kisik i dušik izbijajući elektrone iz njihovih atoma. Tako nastaje **ionosfera**, područje relativno velike koncentracije električki nabijenih čestica – iona i elektrona. U ionosferi se pojavljuje polarna svijetlost. Ionosfera također utječe i na širenje radiovalova.

3.3 Atmosferski tlak

Molekule plinova u zraku neprekidno se i nepravilno gibaju te stoga sa svih strana udaraju u predmete koji su u dodiru sa zrakom. Udarci su tako česti da djeluju kao neprekidna sila. Ta sila, podijeljena s površinom na koju okomito djeluje, jest atmosferski tlak ili tlak zraka. Tlak na izloženu plohu jednak je s obiju njezinih strana, bila ta ploha vodoravna, uspravna ili kosa. Atmosferski tlak se iskazuje u hektopaskalima (hPa), a dopuštena je i upotreba izvansustavne jedinice mbar. Po vrijednosti hektopaskal odgovara milibaru. U standardnoj atmosferi, koja se kao model rabi u raznim proračunima, tlak na razini mora iznosi 1013 hPa.

Atmosferski tlak smanjuje se s porastom visine isprva naglo, a zatim sve sporije (Slika 1). Smanjenje tlaka ovisi i o temperaturi zraka, te je u toplom zraku sporije nego u hladnome. Promjene tlaka s visinom tolike su da barometri već na različitim katovima zgrada ne pokazuju jednake vrijednosti. Primjerice, u blizini Zemljine površine povećanje visine za otprilike 8 m uzrokuje smanjenje tlaka za 1 hPa, a jednaku promjenu tlaka na 5 km visine uzrokuje povećanje visine od oko 15 m.

3.4 Toplinska energija u atmosferi

Zemljina površina prima energiju upijanjem Sunčeva zračenja (danju) i atmosferskog zračenja (danju i noću), a neprekidno je gubi vlastitim zračenjem. Mjerenja u jednakim vremenskim odsječcima danju i noću pokazuju da je danju veći primitak, a noću je obično veći gubitak zračenja. Zbog toga se Zemljina površina danju grije, a noću hladi. U doba godine kad noći postaju kraće, primljena je energija iz dana u dan sve veća i temperatura Zemljine površine raste. Naprotiv, u razdoblju kad su dani kratki primljena je energija malena, pa se na takvim mjestima Zemljina površina sve više hladi.

Premda je zrak loš vodič topline, njegov najniži sloj, debeo nekoliko centimetara, kondukcijom razmjenjuje toplinu s podlogom. Istodobno u razmjeni topline zračenjem sudjeluje mnogo deblji sloj zraka, osobito ako obiluje vlagom i ugljik – dioksidom. Vjetar će miješati zrak koji prima temperaturu podloge sa zrakom iz daljine ili iz visine. Tako velike količine zraka iz troposfere sudjeluju u razmjeni topline konvekcijom. Ako je vjetar slab ili ga nema, uz tlo nastaje danju (i osobito ljeti) sloj ugrijanog zraka, a noću (i osobito zimi) sloj ohlađenog zraka. Za daljnju razmjenu topline danju je bitna konvekcija, a noću kondukcija i zračenje. Naime, pri tlu ugrijan zrak diže se zbog uzgona, a na njegovo mjesto dolazi hladniji iz višeg sloja. On se također grije, diže itd. pa nastaje vertikalno miješanje zraka, što ga obično zovemo *toplinska konvekcija*. Promjer nastalih vrtloga kreće se od nekoliko centimetara do nekoliko stotina metara, a ovisi o atmosferskoj stabilnosti. Takvi vrtlozi mogu podignuti gornju granicu planetarnog graničnog sloja čak iznad 2 km. Noću nema razloga za vertikalno miješanje jer se hladni zrak, koji je ujedno najgušći, već nalazi pri tlu, i samo se na kosom terenu može još spuštati i skupljati na dnu udubine.

Troposfera se grije i hladi od svoje podloge to jače što je temperatura podloge viša, odnosno niža. Budući da oceani imaju mnogo stalniju površinsku temperaturu nego kopno, ni u zraku iznad oceana temperaturne opreke između dana i noći ili ljeta i zime nisu velike.

Primorska klima ima blage zime i ne pretopla ljeta u usporedbi s klimom na kopnima, gdje su zime oštre, a ljeta vruća.

Temperatura zraka odražava odnos između primitka i gubitka toplinske energije, a činitelji o kojima ovisi temperatura na nekome mjestu mogu se razvrstati u tri skupine:

- Visina Sunca i duljina dana pravilno se mijenjaju i od toga potječu pravilne izmjene temperature tijekom dana i godine te razlike između temperature u malim i velikim zemljopisnim širinama. Tako 2 m iznad tla dnevni hod temperature postiže minimum neposredno nakon izlaska Sunca, a maksimum doseže oko dva sata pošto je Sunce bilo u najvišoj točki svoje dnevne staze. Potrebno je, naime, neko vrijeme da se temperatura površine tla odrazi na temperaturi zraka u toj, za meteorologiju standardnoj visini.
- Obilježja Zemljine površine u okolini djeluju stalno ili u duljem razdoblju na jednak način i zbog njih se temperature zraka na različitim mjestima razlikuju. Tako je temperatura na otocima stalnija i blaža nego na kopnu, na planinama niža nego u nizini, a na dnu kotline noći su hladnije nego u ravnici.
- Oblaci, vjetar, horizontalno pomicanje velikih masa zraka i slične pojave uzrok su nepravilnim promjenama temperature, kad, primjerice noću iznenada zatopli ili kad usred ljeta zahladi. Pri računanju srednjih dnevnih i godišnjih hodova temperature iz višegodišnjih mjerenja ponište se nepravilne promjene, pa su krivlje srednjeg dnevnog i godišnjeg hoda glatke.

3.5 Voda u atmosferi

Atmosfera sadrži samo 0,001% ukupne vode na Zemlji, i taj je dio u neprekidnom gibanju. U zrak ulazi isparavanjem s podloge kao nevidljiva para, a izlazi u obliku oborine koja ili pada iz oblaka ili se izravno taloži pri tlu. U atmosferi je zračne struje raznose na velike udaljenosti, a zajedno s njom i toplinu u skrivenome, latentnom obliku. U povoljnim okolnostima voda mijenja agregatno stanje, pri čemu se latentna toplina oslobađa ili troši. Voda u atmosferi sprečava jako ohlađivanje tla i uzrok je mnogim pojavama koje uljepšavaju izgled neba i krajolik.

Na svakoj graničnoj plohi između vode u tekućemu ili čvrstom stanju ili nekoga vlažnog tijela i zraka molekule vode odlaze u zrak kao para, a druge se pak vraćaju iz zraka. Ako je prvih više nego drugih, voda se **isparava**. Na to isparavanje se troši toplina. Ono je

jače pri višoj temperaturi, jačem vjetru i nižoj relativnoj vlažnosti zraka. Ako su te tri veličine poznate, može se, barem približno, izračunati količina isparene vode, uz uvjet da je ima dostatno (*potencijalno isparavanje*). Uz zadanu temperaturu, vjetar i relativnu vlažnost, stvarno će isparavanje biti manje od potencijalnog ako objekt s kojega para odlazi nije dovoljno vlažan. Isparavanje s tla izravno i preko bilja zove se *evapotranspiracija*.

Vlaga u zraku je samo vodena para, a ne kapljice vode niti čestice leda kako se uvriježeno misli. Vodene pare ima u zraku najviše u donjoj troposferi, posebno u blizini velikih vodenih površina. Količina pare u nekom dijelu zraka iskazuje se na više načina:

- Osnovna je mjera **tlak vodene pare**. On je dio ukupnog tlaka zraka (parcijalni tlak). Iskazuje se hektopaskalima, a određuje pomoću psihometra. Što je više vlage u zraku, tlak pare je veći, ali malo kad doseže vrijednost 40 hPa. Za svaku temperaturu zraka postoji najveća vrijednost koju tlak pare može postići. Zovemo je *ravnotežni tlak pare*. Postiže se kad je zrak zasićen vlagom, tj. kad je broj molekula koje prelaze u paru jednak broju molekula koje se vraćaju u tekuću ili u čvrstu fazu. Ravnotežni tlak pare veći je pri višoj temperaturi zraka, a eksponencijalna veza između tog tlaka i temperature obično se prikazuje tablično. Ako je temperatura negativna, ravnotežni je tlak pare nad prehladnom vodom veći nego nad ledom.
- Druga važna mjera je **relativna vlažnost**. Njome se iskazuje koliko je vodene pare u zraku prema najvećoj mogućoj količini pri istoj temperaturi zraka. Definira se naznačenim omjerom tlakova, a iskazuje u postocima:

$$relativna\ vlažnost = \frac{stvarni\ tlak\ pare}{ravnotežni\ tlak\ pare} \cdot 100$$

  Pri zasićenju relativna je vlažnost 100%. Treba primijetiti da se, uz jednaku količinu pare u zraku, relativna vlažnost smanjuje kad temperatura zraka raste, i obrnuto. Zato je dnevni hod relativne vlažnosti često suprotan dnevnom hodu temperature zraka.
- Iako nisu uobičajene, postoje još neke veličine kojima se iskazuje vlažnost zraka. *Omjer miješanja* je masa vodene pare (u gramima) u odnosu prema masi suhog zraka (u kilogramima), a određuje se iz tlaka pare i tlaka zraka. *Specifična vlažnost* izračunava se iz istih podataka i bliska joj je po iznosu. To je masa vodene pare (u gramima) u odnosu prema masi vlažnog zraka (u kilogramima). *Apsolutna vlažnost* ili *gustoća vodene pare* jest masa vodene pare koja se nalazi u jediničnom obujmu zraka

(kg/m^3). Računa se iz tlaka pare i temperature zraka. *Točka rošenja* je temperatura pri kojoj bi, uz postojeću količinu vodene pare, nastalo zasićenje. Određuje se iz tlaka pare, a iskazuje u Celzijevim stupnjevima.

3.6 Atmosferska gibanja

Gibanje zraka jedna je od najvažnijih značajki atmosfere. Posebno u troposferi, gdje se velike i manje česti zraka dižu i spuštaju te koso premještaju, ovise o tom gibanjima svojstva zraka, izgled neba i atmosferske pojave, dakle, ono što u pojedinom trenutku karakterizira vrijeme, a u dugom razdoblju klimu. Najopćenitiji naziv za gibanje zraka jest strujanje. Zatvoreni sustav strujanja naziva se kruženjem ili cirkulacijom, a strujanje paralelno sa Zemljinom površinom zovemo vjetar. Svako je strujanje turbulentno. To znači da čestice zraka ne napreduju isključivo u smjeru strujanja nego se komešaju i nepravilno vijugaju oko njega. Tako se toplinska energija, vlaga i primjese ne raznose samo u smjeru strujanja nego i širom po atmosferi.

4 ELEKTRONIČKA IZVEDBA

Elektronička izvedba meteorološke postaje podijeljena je u dvije cjeline. Prvu čine senzori i sklopovi za obradu signala te sklop za prikupljanje i slanje tih signala, a drugu sklop za prihvat i obradu signala te prikazivanje primljenih parametara na računalu i LCD ekranu.

Osnovni princip rada meteorološke postaje je sljedeći:

- očitavanje stanja atmosfere pomoću senzora
- prilagodba signala radi kompatibilnosti sa mikrokontrolerom
- prikupljanje svih dobivenih signala te njihovo direktno slanje prema ekspanderu mikrokontrolerskog modula
- primanje signala i slanje na unaprijed definirane pinove mikrokontrolera
- obrada prikupljenih podataka i izračunavanje stvarnih vrijednosti
- prikaz dobivenih vrijednosti

Važno je napomenuti da se svi signali senzora prikupljaju u takozvanom ekspanderu senzorskih krugova koji kao izlaz ima 15 – pinski troredni ženski konektor te se isti 15-žilnim kabelom spaja na ekspander mikrokontrolerskog modula, kojim se signali dovode do odgovarajućih pinova mikrokontrolera.

U ovom će poglavlju biti opisane karakteristike senzora, prilagodba izlaznih signala, tok signala te sama arhitektura AT89C51ID2 mikrokontrolera i osnovni princip spajanja elektroničkih komponenata za njegov pravilan rad.

4.1 Mjerna osjetila i prilagodba signala

Pri odabiru mjernih osjetila (senzora) najveću ulogu je imala njihova dostupnost na tržištu. Tako su svi senzori kupljeni preko tvrtke „Altpro.d.o.o.“ koja je generalni zastupnik tvrtke „Farnell“ za Hrvatsku. Sljedeći ključan faktor bila je inteligencija senzora u odnosu na cijenu. Jedini senzor koji nije kupljen je magnetski senzor koji služi za indikaciju brzine vjetra. On je izvađen iz disketne jedinice računala, a imao je ulogu mjerenja brzine vrtnje same magnetske trake diskete.

Za svaki senzor, u programskom paketu *Protel*, projektirane su elektroničke pločice. Izlazni pinovi svake pločice su kvadratni strip pinovi koji se žičanom vezom povezuju sa ekspanderom senzorskih krugova.

4.1.1 Senzor za indikaciju smjera vjetra

Za indikaciju smjera vjetra upotrijebljen je apsolutni mehanički enkoder 25LB22-H (Slika 2) tvrtke Grayhill. Osnovne karakteristike ovog senzora su kako slijedi:

- Struja preklapanja – 150 mA pri istosmjernom naponu od 14V
- Maksimalna struja – 250 mA pri istosmjernom naponu od 24V
- Otpor kontakata – 75 mΩ
- Otpor izolacije – 1000 mΩ između izlaznih pinova
- Napon proboja – 1000 V izmjeničnog napona između izlaznih pinova
- Radna temperatura i temperatura skladištenja – od -65°C do +85 °C
- Predviđeni radni vijek – 1 000 000 preklapanja

Slika 2. Apsolutni enkoder 25LB22-H

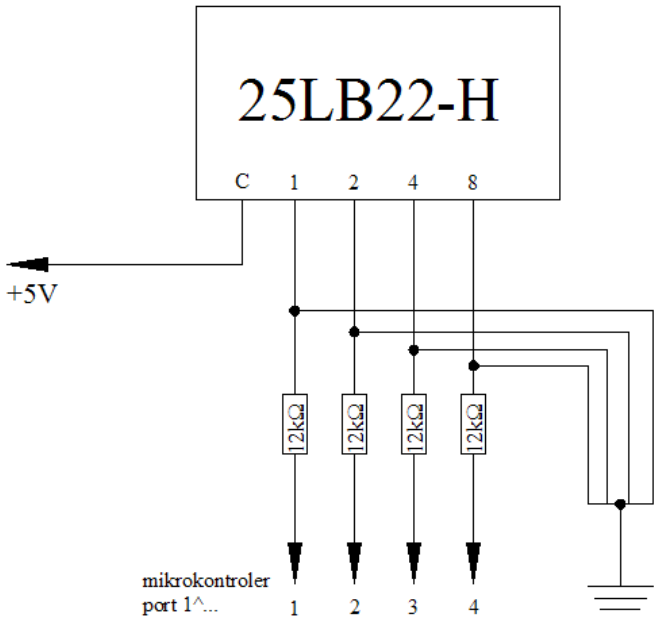
Izlaz senzora je 4 – bitni heksadecimalni broj. Izlazni signali senzora prikazani su u tablici (Tablica 2).

Tablica 2. Tablica istine apsolutnog enkodera 25LB22-H

Rotacija u smjeru kazaljke na satu					
4 - bitni heksadecimalni kod - 16 pozicija					
Pozicija	Izlazni kod	Izlaz 1	2	4	8
1	0				
2	1	●			
3	2		●		
4	3	●	●		
5	4			●	
6	5	●		●	
7	6		●	●	
8	7	●	●	●	
9	8				●
10	9	●			●
11	10		●		●
12	11	●	●		●
13	12			●	●
14	13	●		●	●
15	14		●	●	●
16	15	●	●	●	●

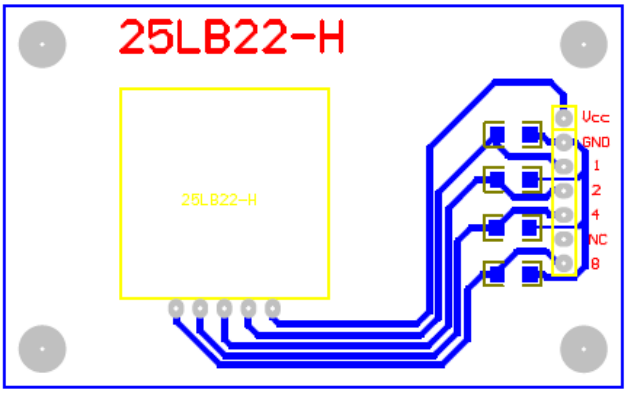
Izlazni signal senzora definiraju četiri mehaničke sklopke koje su njegov sastavni dio. Samim time je vidljivo da postoji velika mogućnost upotrebe takve komponente, tako da je za specifičnu upotrebu potrebno modelirati odgovarajući strujni krug.

Sam senzor ima 5 pinova, od kojih je jedan zajednički (*C*), a ostali su izlazni (od 1, 2, 4, 8). Radi jednostavnijeg programiranja, zajednički pin će biti spojen na napon napajanja od 5 V, dok će izlazni pinovi, preko dijelila napona od 12 kΩ, biti spojeni na ulazne pinove mikrokontrolera (Slika 3). Time su usklađeni izlazni naponi senzora s mikrokontrolerom.



Slika 3. Shema spajanja apsolutnog enkodera

Poznavajući elektroničku shemu spajanja, projektirana je elektronička pločica za apsolutni enkoder (Slika 4).



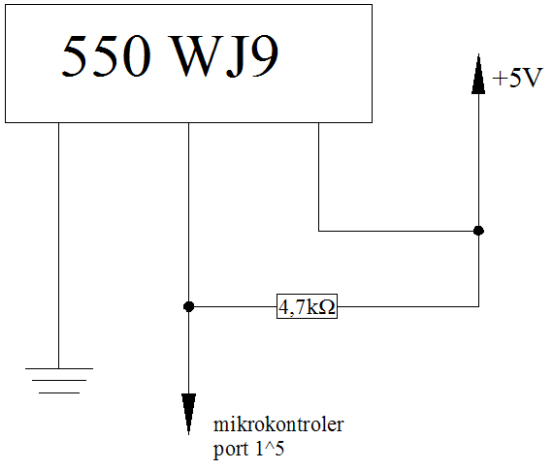
Slika 4. Elektronička pločica za apsolutni enkoder

4.1.1 Senzor brzine vjetra

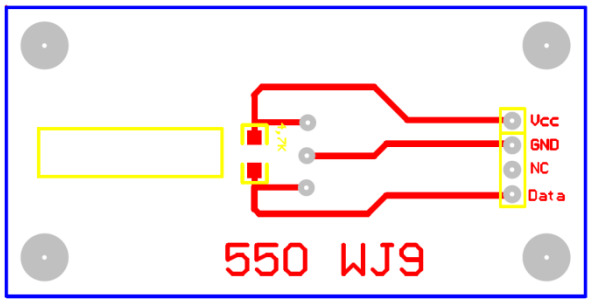
Za indikaciju brzine vjetra koristi se Halle-ov magnetski senzor „550 WJ9“. Neodimijski magnet služi kao pobudni element, pri čemu senzor mijenja svoje stanje.

Kako je navedeni senzor preuzet iz disketne jedinice računala, nije postojalo puno informacija o njemu. Zato se eksperimentalno trebala odrediti maksimalna frekvencija rada, raspored pinova te ulazni i izlazni napon. Poznavanjem maksimalnog broja okretaja disketne jedinice, sa sigurnošću se dalo zaključiti da je frekvencija rada najmanje 10 Hz. Kako je to za vjetar od 30 m/s premala vrijednost, bilo je potrebno provjeriti je li to maksimalno. Vrtnjom improvizirane naprave na kojoj se nalazio magnet izmjereno je 24 impulsa u sekundi, što je i više nego dovoljno. Može se zaključiti da je brzina rada senzora i veća, ali u ovom slučaju to nije potrebno.

Raspored pinova senzora, kao i ulazni napon određeni su praćenjem toka signala na elektroničkom dijelu disketne jedinice. Za tu svrhu bilo je potrebno poznavanje rada sa osciloskopom te multimetrom. Od tri pina senzora, ustanovljeno je da jedan služi za spajanje na ulazni napon od 5 V, drugi na masu, a treći je izlaz senzora, koji se vodi na ulaz mikrokontrolera. Također, ustanovljeno je da je logika senzora obrnuta, tj. kada nema signala on je u logičkoj jedinici, a kada je signal prisutan, spušta se u logičku nulu. Sljedeće slike predstavljaju elektroničku shemu spajanja (Slika 5) te projektiranu elektroničku pločicu (Slika 6).



Slika 5. Shema spajanja 550 WJ9



Slika 6. Elektronička pločica senzora 550 WJ9

4.1.2 Senzor tlaka

Kao osjetilo tlaka koristi se senzor ASDX015A24R proizvođača SenSym. Spada u seriju ASDX koja pokriva širok spektar vrijednosti tlaka te načina mjerenja. Senzor na izlazu daje pojačani analogni napon raspona od 0,5 do 4,5 V uz malu potrošnju energije. Kako bi mjerio apsolutni tlak, u sebi ima vakuumsku referencu te je tako dobivena i kompenzacija temperature, jer ista temperatura djeluje i na vakuumsku referencu, kao i na okolišni tlak zraka.

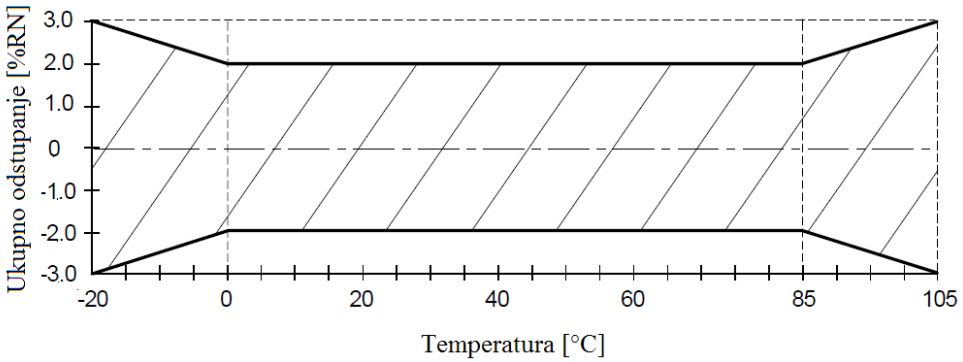
Osjetni raspon tlakova kreće se od 0 do 1034,2 hPa (15 psi), a tlak uništenja senzora je oko 2068 hPa (30 psi). Nazivna osjetljivost iznosi 3,184 mV/hPa. U tablici (Tablica 3) su navedene radne karakteristike senzora.

Tablica 3. Karakteristike senzora tlaka ASDX015A24R

Svojstvo	Min.	Nazivno	Max.	Jedinica
Izlaz pri tlaku 0 hPa	0,42	0,5	0,58	V
Raspon napona (RN)		4		V
Izlaz pri tlaku od 1034 hPa	4,42	4,5	4,58	V
Ukupno odstupanje (od 0 do 85 °C)			±2	%RN
Frekvencija uzorkovanja	100			Hz
Kašnjenje	2,73		14,11	ms
Korak kvantizacije		3		mV
Nazivna struja		6		mA
Napon napajanja	4,75	5	6,5	V
Radna temperatura	-20		105	°C
Temperaturna kompenzacija	0		85	°C

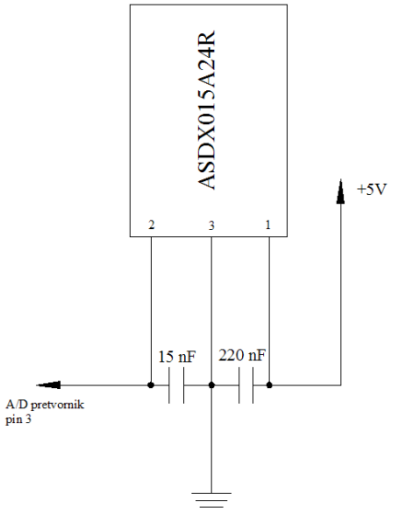
Na slici (Slika 7) je prikazano ukupno odstupanje izmjerene vrijednosti od prave. Vidljivo je da se odstupanje pri temperaturnoj kompenzaciji kreće ±2 %RN, dok pri vrijednostima

temperature kod kojih nema kompenzacije linearno raste, odnosno opada, za ±3 %RN. Također, pri ispitivanju rada senzora ustanovljeno je da je inicijalno odstupanje uvijek jednako 10 hPa, čime se dade zaključiti da vakuumska referenca nije idealna. Zbog toga je softverski potrebno kompenzirati tu grešku dodavanjem +10 hPa na izlaznu vrijednost.



Slika 7. Ukupno odstupanje senzora tlaka

Kako je izlazni signal iz senzora analogni, potrebno ga je što prije digitalizirati. Digitalizacija je potrebna jer otpor vodova smanjuje napon iz senzora što u konačnici može prouzročiti totalno pogrešno izlazno stanje. Na donjoj slici (Slika 8) je vidljivo da izlaz senzora nije spojen na ulazni pin mikrokontrolera, već na ulazni pin A/D pretvornika.



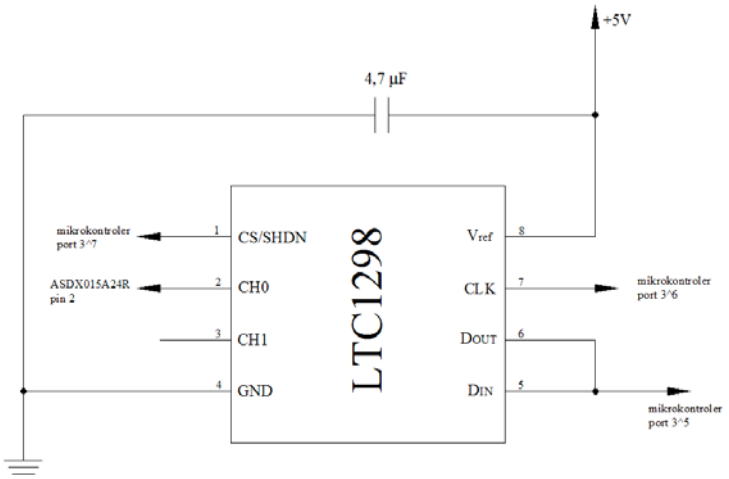
Slika 8. Shema spajanja senzora tlaka ASDX015A24R

4.1.2.1 A/D pretvornik LTC1298

LTC1298 je 12 – bitni 2 - kanalni A/D pretvornik koji radi na principu sukcesivne aproksimacije. Potrebna struja pri A/D pretvorbi iznosi 250 µA, a kada je u stanju pripravnosti, ona opada na 1 nA. Potreban napon napajanja iznosi od 5 do 9 V.

Osnovne značajke LTC1298 A/D pretvornika su sljedeće:

- 12 – bitna rezolucija
- 8 – pinsko DIL kućište
- niska cijena
- mala potrošnja struje od 250 µA
- napon napajanja od 5 do 9 V
- vrijeme pretvorbe od 60 µs
- 2 – kanalni multiplekser

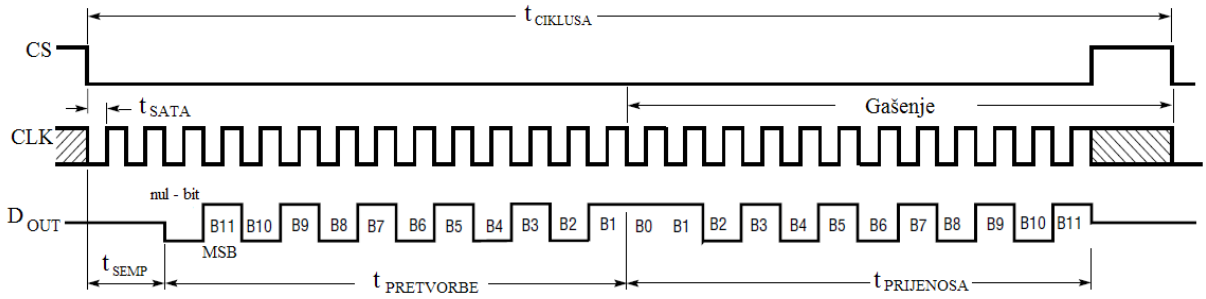


Slika 9. Shema spajanja A/D pretvornika

Tablica 4. Značenje pinova A/D pretvornika

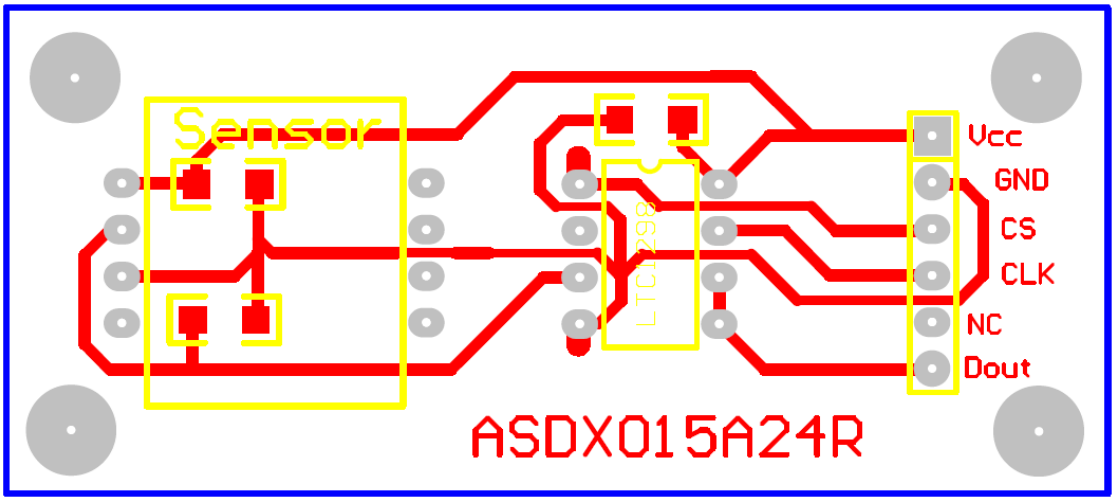
Pin	Značenje
CS/SHDN	Pozivanje pretvornika. Logička nula uključuje, a jedinica isključuje pretvornik
CH0	Analogni ulaz
CH1	Analogni ulaz
GND	Masa
Din	Ulaz digitalnog podatka
Dout	Izlaz digitalnog podatka
CLK	Sat. Sinkronizira prijenos podataka i određuje brzinu prijenosa
Vref	Napon napajanja i referentni napon.

Radi lakšeg praćenja principa rada i prijenosa podataka, u gornjoj tablici (Tablica 4) je prikazano značenje pojedinih pinova pretvornika. CLK sinkronizira prijenos podataka tako da padajući brid spuštanja signala i rastući brid dizanja signala tvore jedan ciklus. Početak komunikacije se pokreće spuštanjem CS signala. Nakon toga LTC čeka početni bit. Nakon što on stigne, 3 – bitna ulazna riječ pomiče se na Din ulaz kojim se konfigurira pretvornik i započinje pretvorba. Nakon jednog nul – bita, dobiva se rezultat pretvorbe koji se šalje preko Dout linije. Nakon završetka prijenosa podataka, CS je potrebno postaviti visoko, čime se resetira pretvornik i omogućuje sljedeća pretvorba. Cijeli proces moguće je vidjeti na donjoj slici (Slika 10).



Slika 10. Sekvenca rada A/D pretvornika

Poznavanjem arhitekture i shema spajanja senzora tlaka te A/D pretvornika, moguće je projektirati elektroničku pločicu za mjerenje tlaka zraka, te prilagodbu i slanje signala (Slika 11).



Slika 11. Elektronička pločica senzora tlaka i sklopa za prilagodbu signala

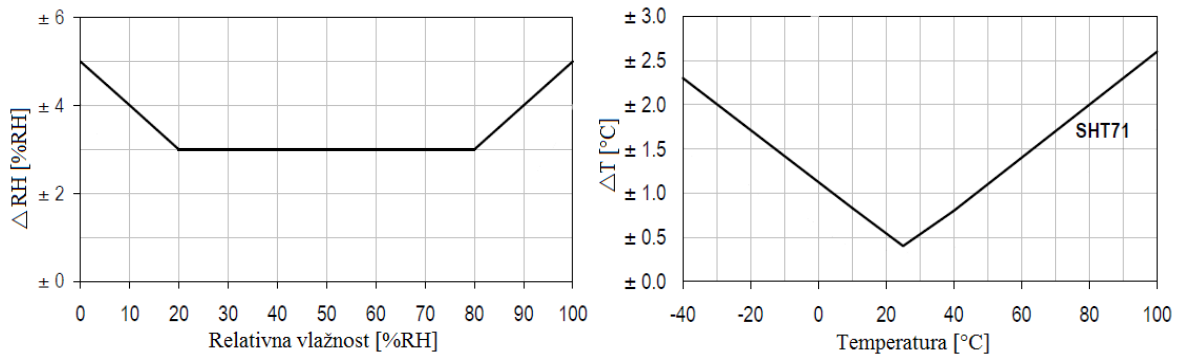
4.1.3 Senzor temperature i relativne vlažnosti

Za mjerenje temperature i vlage upotrjebljen je senzor tvrtke Sensirion, SHT71. U sebi ima integrirana mjerna osjetila te sklop za obradu podataka, čime se dobiva u potpunosti kalibriran digitalni signal na izlazu. Kao osjetilo relativne vlažnosti služi jedinstveni kapacitivni senzor, dok se temperatura mjeri mosnim spojem. Implementirana CMOSens® tehnologija jamči veliku pouzdanost i dugotrajnu stabilnost. Oba mjerna osjetila su direktno upregnta s 14 – bitnim A/D pretvornikom te krugom za serijsku komunikaciju. Time se postigla odlična kvaliteta signala, brzi odziv i neosjetljivost na vanjske poremećaje.

Svaki SHT71 je individualno kalibriran u preciznoj vlago – komori. Zbog toga je bilo moguće upisati sve kalibracijske koeficijente u OTP memoriju. Ti su koeficijenti naknadno služili za internu kalibraciju signala senzora. 2 – žično serijsko sučelje i unutrašnja regulacija napona omogućuju jednostavnu i brzu implementaciju u svaki elektronički sustav.

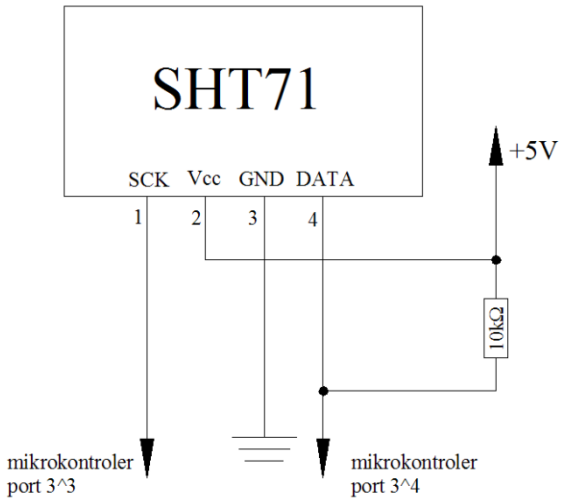
Tablica 5. Karakteristike senzora SHT71

		RELATIVNA VLAŽNOST				**TEMPERATURA**			
Svojstvo	Stanje	Min	Nazivno	Max.	Jedinica	Min	Nazivno	Max.	Jedinica
Rezolucija		0,5	0,03	0,03	%	0,04	0,01	0,01	°C
		8	12	12	bit	12	14	14	bit
Točnost	Nazivna		±3		%		±0,4		°C
	Max.	Slika 12 a)				Slika 12 b)			
Ponovljivost			±0,1		%		±0,1		°C
Histereza			±1		%	---			
Vrijeme odziva	63%τ		8		s	5		30	s
Raspon rada		0		100	%	-40		123,8	°C
Dugoročno odstupanje	Nazivno		<0,5		%/god		<0,04		°C/god



a) b)

Slika 12. Maksimalna točnost a) rel. vlažnosti, b) temperature



Slika 13. Shema spajanja senzora temperature i vlage SHT71

Kao što je vidljivo na slici (Slika 13), senzor ima 4 pina. Pin *SCK* služi za sinkronizaciju komunikacije između senzora i mikrokontrolera. Kako sučelje za komunikaciju ima u potpunosti statičku logiku, ne postoji minimalna frekvencija. Dakako, pri tome se ne smiju zanemariti ostale elektroničke komponente koje mogu omogućiti šumove pri velikoj frekvenciji. *DATA* pin se koristi za prijenos podataka sa i prema senzoru. Da bi se poslao podatak, DATA se detektira na uzdižućem bridu SCK i mora biti stabilan sve dok je SCK visoko. Nakon padajućeg brida SCK, DATA može biti promijenjen. Pin Vcc služi za dovod napona napajanja koji mora biti između 2,4 i 5,5 V.

Komunikacija senzora i mikrokontrolera mora započeti spajanjem napona napajanja, te se mora pričekati minimalno 11 ms prije bilo koje druge akcije. Nakon toga, potrebno je pokrenuti komunikaciju. Početak komunikacije se vrši spuštanjem DATA linije, pri čemu je SCK visoko. Tada slijedi niski impuls na SCK i dizanje DATA linije, pri čemu je SCK

visoko. Sljedeća naredba sastoji se od tri adresna bita (samo 000) i pet opcijskih, kojima se određuje željena aktivnost senzora. Senzor tada označava primitak naredbe te spušta DATA liniju nakon osmog bita SCK, te nazad diže visoko nakon devetog bita SCK.

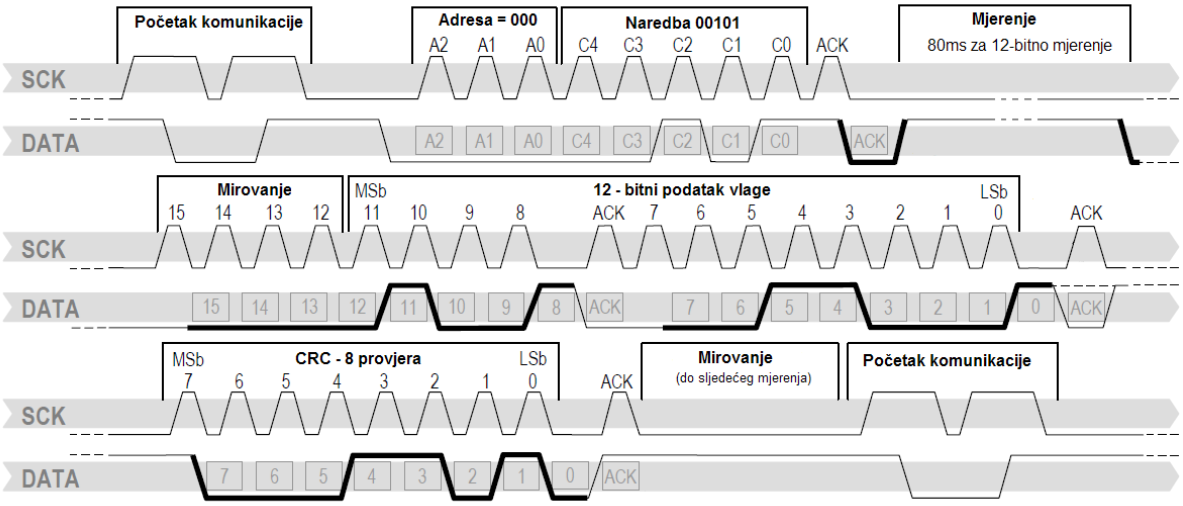
Nakon opcijske naredbe (00000101 za relativnu vlažnosti i 00000011 za temperaturu), mikrokontroler mora sačekati vrijeme potrebno za mjerenje. U slučaju 12 – bitnog mjerenja, vrijeme čekanja iznosi 80 ms. To vrijeme ovisi o brzini unutarnjeg oscilatora, te tako može biti smanjena za 30%. Kao nagovještaj kraja mjerenja, senzor spušta DATA liniju i ulazi u stanje mirovanja. Kontroler obavezno mora sačekati ovaj signal prije resetiranja SCK linije da bi se očitalo izmjereno stanje. Podaci mjerenja su u ovom trenutku spremljeni sve do očitavanja, tako da nije nužno da ih kontroler u istom trenutku mora preuzeti.

Trenutni podatak koji će naknadno biti preuzet od strane kontrolera sastoji se od dva bajta koji predstavljaju izmjerenu vrijednost te jednog bajta CRC podatka (koji služi za provjeru komunikacije). Mikrokontroler preuzima taj paket spuštanjem DATA linije za pojedini bit.

Komunikacija se prekida nakon potvrdnog bita CRC podatka. Ako provjera komunikacije nije indicirana, mikrokontroler može prekinuti komunikaciju nakon preuzimanja podataka mjerenja na način da je ACK visoko. Da bi samozagrijavanje senzora ostalo unutar granice od 0,1 °C, senzor treba ostati u stanju mirovanja minimalno 10% vremena rada.

Ako se desilo da je komunikacija između senzora i kontrolera prekinuta ili se ne može uspostaviti, potrebno je podići DATA liniju visoko, te devet ili više puta preklopiti SCK liniju. Ova akcija mora slijediti nakon početka komunikacije kada se prelazi na novu opcijsku naredbu (s temperature na rel. vlažnost i obrnuto).

Primjer komunikacije i mjerenja prikazan je na donjoj slici (Slika 14). Pretpostavka primjera je da je izlaz senzora jednak vrijednosti 0000 100100110001, što je jednako 2353, tj. preračunato u stvarnu vrijednost, 75,79%. Podebljana DATA linija označava da je kontrolirana od strane senzora, a normalna od strane mikrokontrolera.



Slika 14. Primjer sekvence SHT71 senzora

Da bi se kompenzirala nelinearnost mjernog osjetila vlažnosti, te da se postigne potpuna točnost, preporučljiva je kompenzacija očitane vlažnosti (SO_{RH}) sa sljedećim izrazom, čiji su koeficijenti dani u tablici (Tablica 6):

$$RH_{linearizirano} = c_1 + c_2 \cdot SO_{RH} + c_3 \cdot {SO_{RH}}^3 \quad [\%]$$

Tablica 6. Optimizacijski koeficijenti linearizacije relativne vlažnosti

SO_{RH}	c_1	c_2	c_3
12 – bitni	−2,0468	0,0367	$-1{,}5955 \cdot 10^{-6}$
8 – bitni	−2,0468	0,5872	$-4{,}0845 \cdot 10^{-4}$

Za temperaturu različitu od 25 °C, potrebna je temperaturna kompenzacija dobivene vrijednosti relativne vlažnosti. Spomenutu kompenzaciju moguće je izvršiti prema sljedećem izrazu, čiji su koeficijenti dani u tablici (Tablica 7):

$$RH_{stvarno} = (T_{°C} - 25) \cdot (t_1 + t_2 \cdot SO_{RH}) + RH_{linearizirano}$$

Tablica 7. Koeficijenti temperaturne kompenzacije

SO_{RH}	t_1	t_2
12 – bitni	0,01	0,00008
8 – bitni	0,01	0,00128

Senzor temperature s mosnim spojem izričito je linearne karakteristike. Da bi se digitalni podatak izmjerene vrijednosti (SO_T) pretvorio u stvarnu vrijednost temperature, potrebno ga je izračunati sljedećim izrazom, čiji su koeficijenti dani u tablici (Tablica 8):

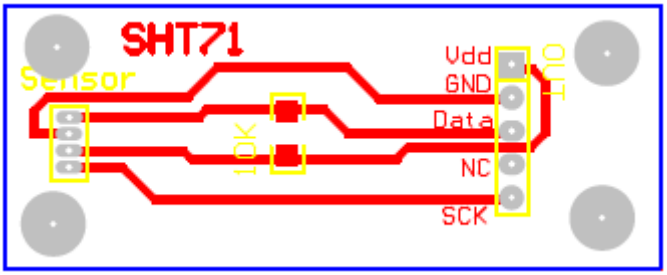
$$T = d_1 + d_2 \cdot SO_T$$

Tablica 8. Koeficijenti pretvorbe temperature

Vdd	d_1
5 V	-40,1
4 V	-39,7
3,5 V	-39,7
3 V	-39,6
2,5 V	-39,4

SO_T	d_2
14 – bitni	0,01
12 – bitni	0,04

Poznavanjem sheme spajanja i svih potrebnih podataka o toku i prilagodbi signala, projektirana je elektronička pločica za senzor temperature i vlage (Slika 15).

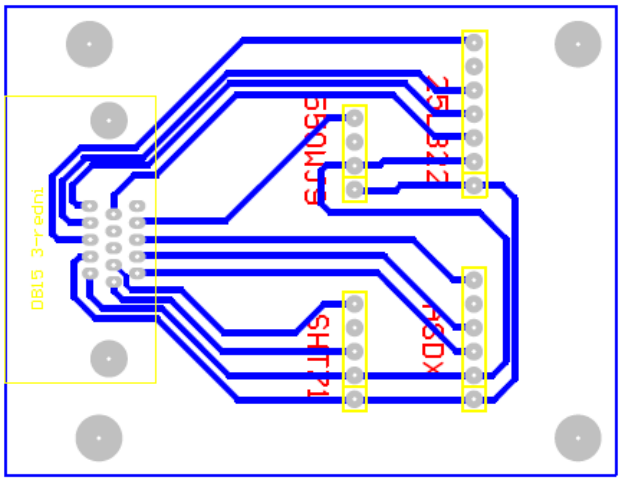


Slika 15. Elektronička pločica senzora temperature i vlage

4.2 Ekspander senzorskih krugova

Ekspander senzorskih krugova nije ništa drugo već elektronička pločica projektirana tako da prikuplja signale sa svih senzora, koji će se tada povezati sa ekspanderom mikrokontrolerskog modula.

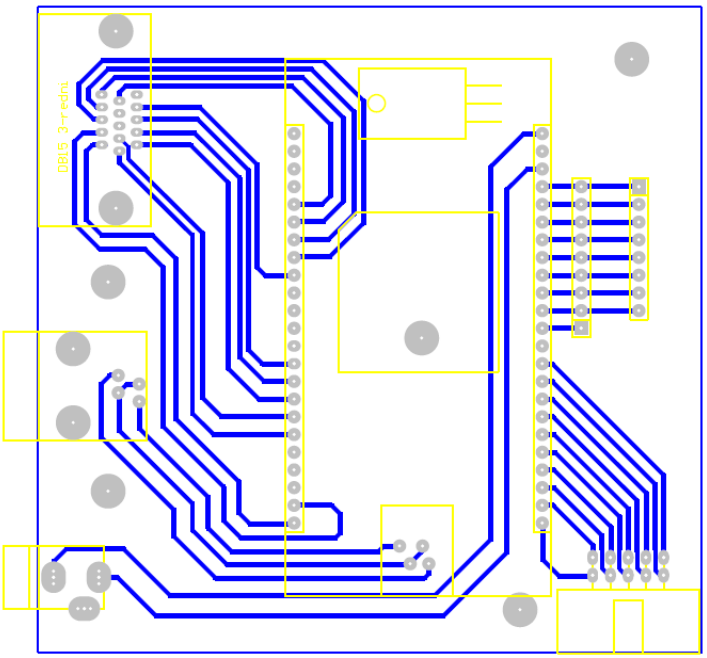
Kako je već ranije spomenuto, elektroničke pločice svakog senzora kao izlaz imaju odgovarajući broj strip muških pravokutnih konektora. Oni se tada višežilnim kabelom spajaju na ekspander, gdje su prikupljeni u DB15 – pinskom ženskom trorednom konektoru. Projektirana elektronička pločica prikazana je na slici (Slika 16).



Slika 16. Ekspander senzorskih krugova

4.3 Ekspander mikrokontrolerskog modula

Nakon što su sve linije sa senzora prikupljene, 15 – žilnim kabelom dovedene su do ekspandera mikrokontrolerskog modula. Njegova je svrha sve signale dovesti na pinove mikrokontrolera. Također, on povezuje drugi port kontrolera sa LCD ekranom preko DB9 priključka, dovodi napon napajanja prema stabilizatoru napona, te preko telefonske utičnice ostvaruje serijsku vezu sa računalom. Na sljedećoj slici (Slika 17) prikazana je njegova elektronička pločica.



Slika 17. Ekspander mikrokontrolerskog modula

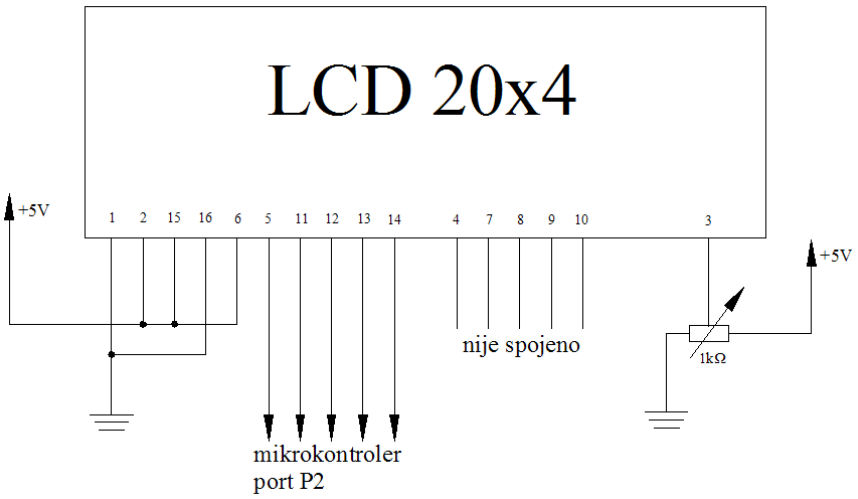
4.4 LCD ekran

Korišteni LCD ekran je proizvod tvrtke Winstar. Veličine je 4 retka x 20 stupaca, kod kojih je veličina točke 0,55x0,55 mm, a visina jednog znaka 2,95x4,75 mm. Nazivni napon napajanja iznosi 5 V, dok je minimalni 4,7 V, a maksimalni 5,5 V.

Ima 16 pinova, čije je značenje prikazano u tablici (Tablica 9).

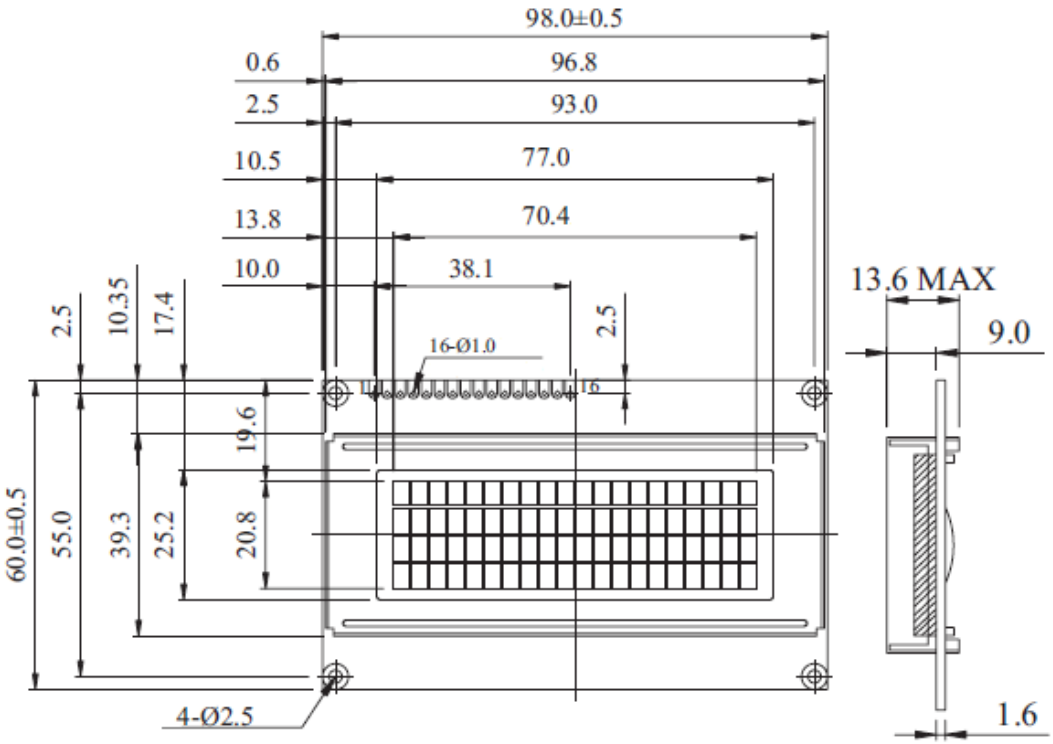
Tablica 9. Raspored pinova LCD ekrana

Redni broj pina	Simbol	Funkcija
1	GND	Masa
2	Vdd	Napon napajanja
3	V0	Podešavanje kontrasta
4	RS	1 – podaci 2 – instrukcije
5	R/W	1 – čitanje 2 – pisanje
6	E	Omogući signal
7	DB0	Podatkovna linija
8	DB1	...
9	DB2	
10	DB3	...
11	DB4	...
12	DB5	...
13	DB6	...
14	DB7	...
15	A	+4,2 V za pozadinsko osvjetljenje
16	K	Masa pozadinskog osvjetljenja



Slika 18. Shema spajanja LCD ekrana

Kako bi bilo moguće projektirati kućište, moraju biti poznate dimenzije ekrana, koje su prikazane na slici (Slika 19).



Slika 19. Dimenzije LCD ekrana

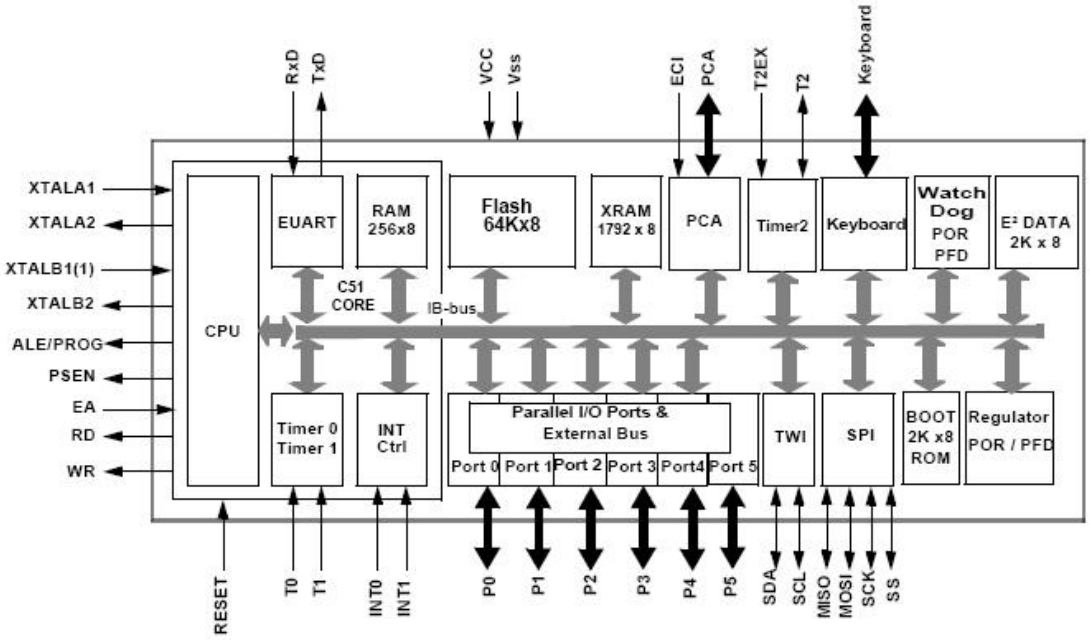
4.5 Mikrokontroler AT89C51ID2 (PCC44)

AT89C51ID2 je kontroler odličnih performansi, CMOS Flash verzija jednočipnog 8 – bitnog 80C51 CMOS kontrolera. Sadrži memorijski blok od 64 kbajta flash memorije za programski kod i spremanje podataka, koja nudi mogućnost programiranja u paralelnom i serijskom modu. Napajanje koje je potrebno za programiranje u cijelosti se dovodi unutar samog kontrolera, i to preko standardnog V_{CC} ulaza. Također, kontroler zadržava sve karakteristike starijeg kontrolera 80C52 kao što su 256 bajta unutarnje RAM memorije, 10 izvora prekida sa 4 stupnja prioriteta te 3 brojila. Nadalje, kontroler ima programibilno polje brojila, 1792 bajta XRAM memorije, hardverski watchdog timer, SPI, univerzalan serijski kanal koji omogućuje multiprocesorsku komunikaciju (EUART), te mehanizam za povećanje brzine (X2). Potpuno statička arhitektura kontrolera nudi mogućnost smanjenja potrošnje energije, smanjujući frekvenciju sata na bilo koju vrijednost, a da se time ne dovede u pitanje mogućnost gubljenja podataka. Također, dodatna svojstva kontroleru omogućuju odličan rad za sustave kojima je potrebna pulsno – širinska modulacija te velika brzina I/O komunikacije pa su zato prikladni kod alarmnih sustava, upravljanja motorima, fiksnim telefonima, čitačima pametnih kartica i gdjegdje drugdje.

4.5.1 Osnovne značajke:

- 80C52 kompatibilnost
- kompatibilnost s 8051 setom naredbi
- četiri 8 – bitnih I/O portova
- tri 16 – bitnih brojila
- 256 bajta RAM memorije
- 10 izvora prekida sa 4 stupnja prioriteta
- ISP (In-System programming) korištenjem standardnog V_{CC} izvora napajanja
- integrirani nadzor napajanja
- arhitektura za brzi rad
- 64 kbajta flash memorije za program ili podatke
- 1792 bajta proširene RAM memorije (XRAM)
- 2048 bajta EEPROM memorije za spremanje podataka
- 32 KHz kristalni oscilator
- SPI sučelje (master/slave mod)

- asinkroni reset portova
- hardverski watchdog timer
- napon napajanja od 2,7V do 5,5V
- industrijski temperaturni opseg rada (-40 do + 85°C)



Slika 20. Blok dijagram AT89C51ID2 kontrolera

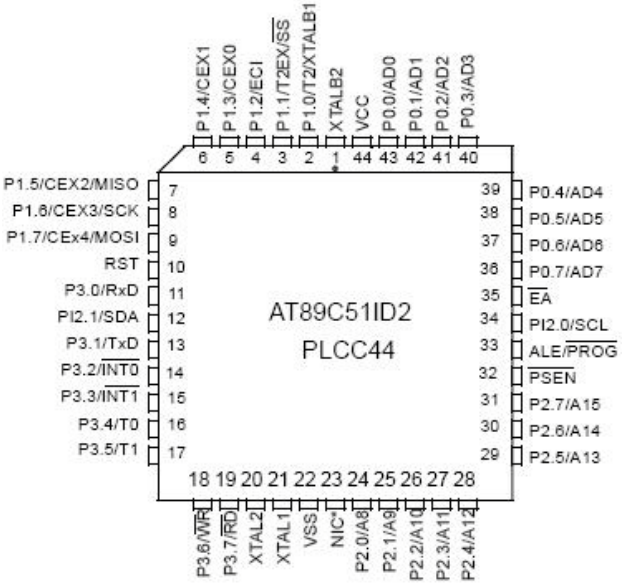
4.5.2 Registri posebne namjene

Registri posebne namjene dijele se na sljedeće kategorije:

- C51 jezgreni registri: ACC, B, DPH, DPL, PSW, SP
- registri I/O portova: P0, P1, P2, PI2
- registri brojila: T2CON, T2MOD, TCON, TH0, TH1, TH2, TMOD, TL0, TL1, TL2, RCAP2L, RCAP2H
- serijski I/O port registri: SADDR, SADEN, SBUF, SCON
- CPA registri: CCON, CCAMPx, CL, CH, CCAPxH, CCAPxL
- kontrolni registar napajanja i sata: PCON
- hardverski watchdog timer registri: WDTRST, WDTPRG
- prekidni registri sustava: IE0, IPL0, IPH0, IE1, IPL1, IPH1
- registri sučelja tipkovnice: KBE, KBF, KBLS
- SPI registri: SPCON, SPSTR, SPDAT
- 2 – žični registri sučelja: SSCON, SSCS, SSDAT, SSADR
- flash registar: FCON

- registar preljeva sata: CKRL
- ostali: AUXR, AUXR1, CKCON0, CKCON1

4.5.3 Konfiguracija pinova i značenje



Slika 21. Konfiguracija pinova AT 89C51ID2

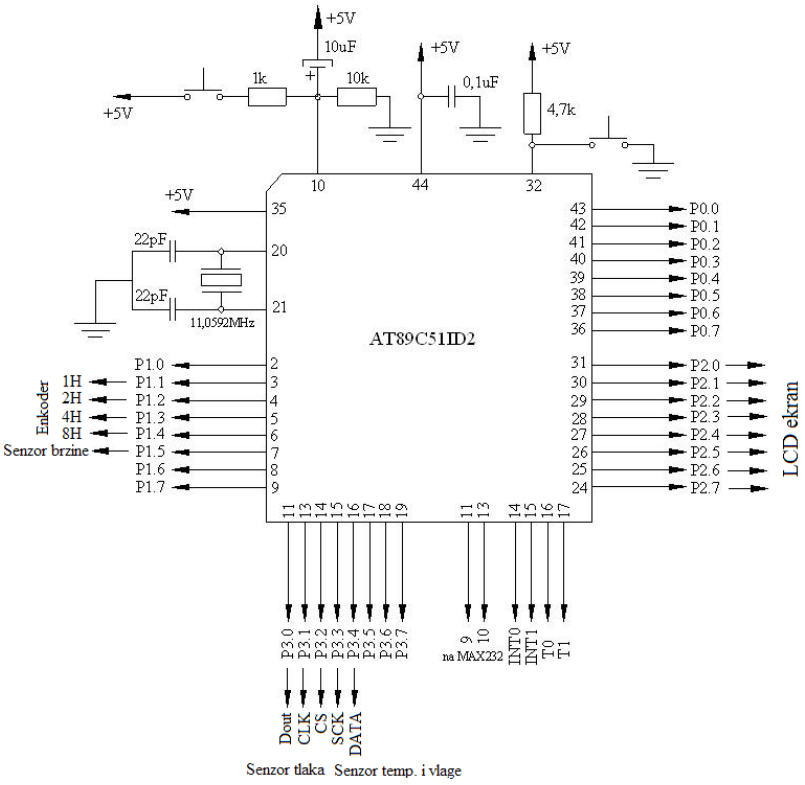
Tablica 10. Značenje pinova mikrokontrolera

Mnemonički	Broj pina	Tip	Ime i funkcija
V_{SS}	22	U	**Uzemljenje: referentno 0V**
V_{CC}	44	U	**Napon napajanja: napajanje za sve modove rada**
P0.0 - P0.7	43 - 36	U/I	**Port 0: dvosmjerni I/O port. Pinovi porta na kojima su upisane jedinice su slobodni i mogu biti korišteni kao ulazi visoke impedancije.**
P1.0 - P1.7	2 - 9	U/I	**Port 1: 8- bitni dvosmjerni port sa unutarnjom promjenom stanja. Pinovi porta kojima su upisane jedinice su dignuti visoko služe kao ulazi.**
	2	U/I U/I U	**P1.0: ulaz/izlaz** **T2 : brojilo 2 – vanjsko brojenje** **XTALB1: ulaz pomoćnog sata na invertirajuće pojačalo oscilatora**
	3	U/I U U	**P1.1: ulaz/izlaz** **T2EX: brojilo 2- reset/stop/smjer konrtola** **$\overline{SS}$: SPI slave odabir**

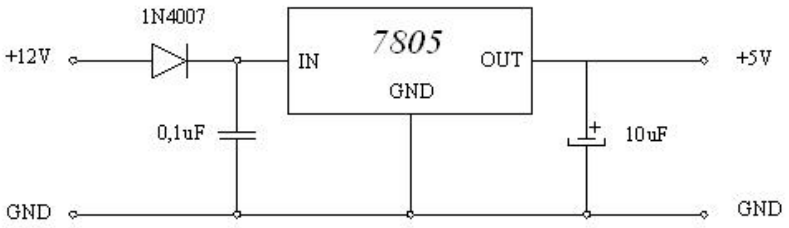
	4	U/I U	**P1.2: ulaz/izlaz** **ECI: vanjski sat za PCA**
	5	U/I U/I	**P1.3: ulaz/izlaz** **CEX0: hvatanje/usporedba vanjskog I/O za PCA modul 0**
	6	U/I U/I	**P1.4: ulaz/izlaz** **CEX1: hvatanje/usporedba vanjskog I/O za PCA modul 1**
	7	U/I U/I U/I	**P1.5: ulaz/izlaz** **CEX2: hvatanje/usporedba vanjskog I/O za PCA modul 2** **MIS0: SPI master ulaz, slave izlaz**
	8	U/I U/I U/I	**P1.6: ulaz/izlaz** **CEX3: hvatanje/usporedba vanjskog I/O za PCA modul 3** **SCK: SPI serijski sat**
	9	U/I U/I U/I	**P1.7: ulaz/izlaz** **CEX4: hvatanje/usporedba vanjskog I/O za PCA modul 4** **MIS1: SPI master ulaz, slave izlaz**
XTALA1	21	U	**Kristal A 1: ulaz na invertno pojačalo oscilatora i ulaz na krugove unutarnjeg generatora sata**
XTALA2	20	I	**Kristal A 2: izlaz sa invertnog pojačala oscilatora**
XTALB1	2	U	**Kristal B 1: (pomoćni sat) ulaz na invertno pojačalo oscilatora i ulaz na krugove unutarnjeg generatora sata**
XTALB2	1	I	**Kristal B 2: (pomoćni sat) izlaz sa invertnog pojačala oscilatora**
P2.0 – P2.7	24 – 31	U/I	**Port 2: 8- bitni dvosmjerni port sa unutarnjom promjenom stanja. Pinovi porta kojima su upisane jedinice su dignuti visoko i služe kao ulazi. Važno je napomenuti da prilikom pristupanja vanjskoj podatkovnoj memoriji koja koristi 8- bitne adrese, port 2 odašilje sadržaj registra posebne namjene P2**
P3.0 – P3.7	11, 13 - 19	U/I	**Port 3: 8- bitni dvosmjerni port sa unutarnjom promjenom stanja. Pinovi porta kojima su upisane jedinice su dignuti visoko i služe kao ulazi.**
	11	U	**RXD: serijski ulazni port**
	13	I	**TXD: serijski izlazni port**
	14	U	**$\overline{\text{INT0}}$: vanjski prekid 0**
	15	U	**$\overline{\text{INT1}}$: vanjski prekid 1**
	16	U	**T0: vanjski prekid brojila 0**
	17	U	**T1: vanjski prekid brojila 1**
	18	I	**$\overline{\text{WR}}$: signal za pisanje u vanjsku podatkovnu memoriju**
	19	**I**	**$\overline{\text{TR}}$: signal za čitanje iz vanjske podatkovne memorije**

4.5.4 Spajanje osnovnih elemenata

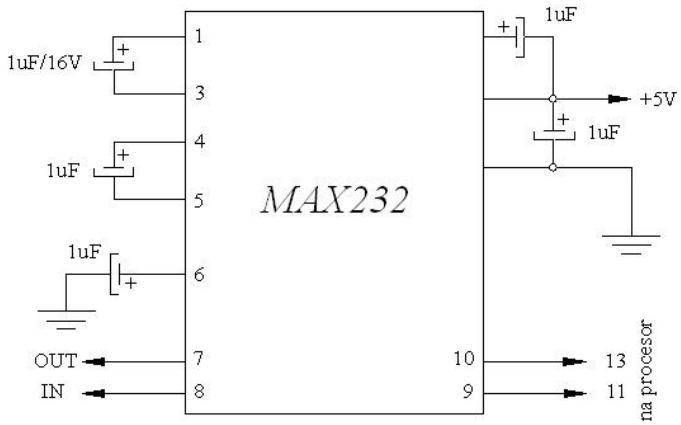
Osnovni elementi za rad kontrolera su kristalni oscilator, stabilizator napona (7805), te sklop za prilagodbu signala za serijsku komunikaciju s računalom (MAX232). Na sljedećim slikama prikazane su sheme spajanja pojedinih navedenih elemenata.



Slika 22. Shema spajanja osnovnih elemenata na AT89C51ID2

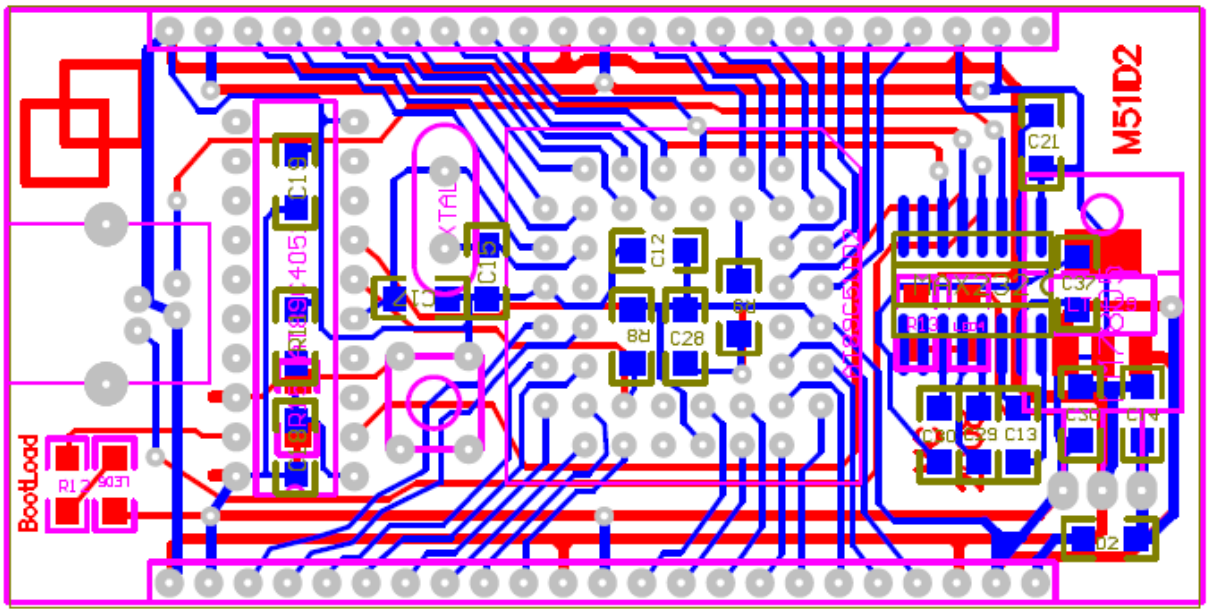


Slika 23. Shema spajanja stabilizatora napona 7805



Slika 24. Shema spajanja sklopa za serijsku komunikaciju MAX232

Poznavanjem električnih shema spajanja svih elemenata, projektirana je elektronička pločica za mikrokontroler (Slika 25).



Slika 25. Elektronička pločica mikrokontrolera AT89C51ID2

5 PROGRAMSKA IZVEDBA

Programska izvedba podijeljena je u dva dijela. Prvi dio je izrada računalnog sučelja koje omogućuje praćenje meteoroloških parametara na računalu, te njihovo spremanje. Sučelje je izvedeno tako da i korisnicima koji nemaju veliko znanje o računalima omogući lagano praćenje atmosferskih prilika. Ovaj prvi dio programiran je u objektno orijentiranom jeziku Delphi7. Drugi stupanj programiranja svodi se na programiranje kontrolera. Ono je izvedeno u klasičnom programskom jeziku C, a služi za primanje i obradu podataka sa senzora te njihovo prikazivanje na LCD ekranu te slanje na računalo koristeći serijsku vezu.

U ovom dijelu biti će najprije opisano objektno orijentirano programiranje te kratak opis programa Delphi7. Nakon toga će biti navedene često korištene funkcije u razvijenoj aplikaciji simboličnog naziva „Meteo“. U nastavku biti će opisane najčešće korištene funkcije pri programiranju kontrolera te zahtjevi koji su postavljeni za ispravan rad.

5.1 Objektno orijentirano programiranje

Dva su osnovna dijela programiranja: podaci i naredbe. Za rad s podacima potrebno je razumijevanje varijabli i tipova, a za rad s naredbama potrebno je razumjeti upravljačke strukture (control structures) i podprograme (subroutines). Varijabla je samo lokacija u memoriji (ili nekoliko lokacija promatranih kao jedinica) kojemu je dodijeljeno ime da ga se u programu može lako pozivati i koristiti. Programer treba voditi računa samo o imenu, vođenje računa o memorijskom mjestu dužnost je kompajlera. Programer ne smije gubiti iz vida da ime upućuje na neku vrstu mjesta u memoriji koje može spremiti podatke, čak i ako ne mora znati gdje je to točno u memoriji. U većini programskih jezika varijabla ima tip (type) koji ukazuje na vrstu podataka koje može spremiti. Jedna vrsta varijabli može sadržavati integer - cjelobrojne vrijednosti, dok druga sadrži floating point - brojeve s decimalnim mjestima. Računalo pravi razliku između integer 1 i floating point 1.0, zapravo u računalu izgledaju sasvim različito. Također postoje i tipovi za pojedinačne znakove, nizove znakova, kao i za manje uobičajene tipove kao što su datumi, boje, zvukovi ili bilo koji drugi tip podataka koje bi program mogao spremiti.

Program je niz naredbi. U običnom odvijanju računalo izvodi ove naredbe redom kako se pojavljuju jednu za drugom. Ovo je očito vrlo ograničen način jer bi računalo vrlo brzo ostalo bez naredbi koje treba izvršiti. Upravljačke strukture su posebne naredbe koje mogu izmijeniti tok odvijanja programa. Postoje dvije osnovne vrste upravljačkih struktura: petlje, koje omogućavaju ponavljanje niza naredbi i grananja koja omogućuju računalu da odluči

između više različitih postupaka ispitivanjem uvjeta koji se javljaju za vrijeme izvršavanja programa. Veliki programi su tako složeni da bi ih bilo gotovo nemoguće napisati bez da ih se "razbije" u lakše ostvarive dijelove. Podprogrami su jedan od načina ostvarivanja tog «razbijanja». Podprogram se sastoji od naredbi za izvršavanje nekog zadatka okupljenih u cjelinu s imenom. Ime se kasnije koristi kao zamjena za čitav niz naredbi. Na primjer, ako je jedan od zadataka koje program mora izvršiti ispisivanje temperature na ekranu, potrebno je naredbe okupiti u podprogram i dati mu prikladno ime, npr. „IspisTemp()“ Nakon toga, na bilo kojem mjestu u programu gdje je potrebno ispisati temperaturu dovoljno je napisati jednu naredbu:

```
IspisTemp();
```

Ovo će imati učinak jednak kao ponavljanje svih naredbi za ispisivanje temperature na svakom mjestu. Prednost nije samo u tome da je manje pisanja. Organiziranje programa u podprograme također pomaže organiziranju razmišljanja i napora u razvoju programa. Za vrijeme pisanja podprograma za ispis temperature moguće se koncentrirati isključivo na taj problem, bez razmišljanja o ostatku programa. Jednom kad je podprogram gotov, može se zaboraviti na detalje – taj problem je riješen. Podprogram postaje kao ugrađeni dio jezika koji je moguće koristiti bez razmišljanja o tome što se događa unutar podprograma. Varijable, tipovi, petlje, grananja i podprogrami su osnova onog što bi se moglo nazvati tradicionalnim programiranjem. Osim toga, kako programi rastu javlja se potreba za dodatnim strukturama za rješavanje njihove složenosti. Jedno od najučinkovitijih sredstava je objektno programiranje.

Objektno orijentirano programiranje je pokušaj približavanja modela programa načinu ljudskog razmišljanja. Kod starog načina programiranja, programer je morao pronaći računalnu zadaću koju treba izvršiti da bi se riješilo neki problem. Programiranje se zatim sastojalo od pronalaženja niza naredbi koje bi izvršile taj zadatak. U osnovi objektno orijentiranog programiranja, umjesto zadaća nalazimo objekte - jedinice koje imaju svoje ponašanje, drže podatke i mogu međusobno djelovati. Programiranje se sastoji od oblikovanja skupa objekata koji na neki način opisuju problem koji treba riješiti. Softverski objekti u programu predstavljaju stvarne ili zamišljene jedinice u području problema. Na ovaj način razvoj programa trebao bi biti prirodniji i stoga lakši za postavljanje i razumijevanje. Do neke granice, objektno orijentirano programiranje je samo promjena točke gledanja, u okvirima standardnog programiranja objekt se može promatrati kao skup varijabli i nekih podprograma za upravljanje tim varijablama. Zapravo, moguće je koristiti objektno orijentirano programiranje u bilo kojem programskom jeziku. Naravno postoji velika razlika između

programskih jezika u kojima je objektno orijentirano programiranje moguće i onih koji ga uistinu aktivno podržavaju. Objektno orijentirani programski jezici poput Delphi7 imaju niz mogućnosti koje ih čine različitim od standardnih programskih jezika. Ispravno razmišljanje je osnova za korištenje tih mogućnosti.

Temelji objektnog programiranja jesu:

- Klase (Classes) - korisnički definiran tip podataka, a sadrži neke interne podatke i metode u obliku procedura i funkcija
- Objekti (Objects) - varijabla tipa definirane klase, odnosno to su stvarni entiteti, ili instance određene klase
- Svojstva (Properties) - deklariranje karakteristika objekata, odnosno svojstva i reakcije vezane uz objekte

Objekti su usko vezani uz klase. Potrebno je najprije razjasniti razlike između objekata i klasa. Klase, zapravo njihovi nestatički dijelovi opisuju objekte. Uobičajeno je reći da objekti pripadaju klasama. Sa programerskog gledišta najtočnije bi bilo reći da se klase koriste za opisivanje objekata. Nestatički dijelovi klasa određuju koje će varijable i potprogrami biti sadržani u objektima. Ovo je ujedno i dio objašnjenja po čemu se objekti razlikuju od klasa: objekti se stvaraju i uništavaju tijekom izvođenja programa, a korištenjem jedne klase može biti stvoren neograničen broj objekata.

Objekt koji pripada klasi naziva se instanca te klase. Varijable koje taj objekt sadrži nazivaju se varijable instance. Potprogrami koje objekt sadrži nazivaju se metode instance. Očito je da su statički i nestatički dijelovi klase vrlo različite stvari i sa vrlo različitim namjenama. Mnogo klasa sadrži isključivo statičke ili nestatičke članove, iako je moguće miješanje statičkih i nestatičkih članova u jednoj klasi. Statičke varijable i potprogrami u klasama se zovu varijable klase, odnosno metode klase, jer pripadaju klasi a ne instancama te klase. Deklariranje varijable ne stvara objekt. Vrlo je važno istaknuti da nikakva varijabla ne može sadržavati objekt, varijabla može sadržavati samo poziv objekta. O objektima treba razmišljati kao da slobodno lebde negdje u memoriji računala. Zapravo, postoji dio memorije u koji se smještaju objekti. Umjesto da sadržava objekt, varijabla sadrži samo podatke neophodne za pronalaženje objekta u memoriji. Ta informacija zove se poziv (reference) ili pokazivač (pointer) na objekt. Zapravo, poziv na objekt je adresa memorijske lokacije u kojoj je spremljen objekt. Pri korištenju varijabli tipa klase računalo koristi poziv u varijabli za pronalaženje stvarnog objekta. Objekti se stvaraju korištenjem odgovarajućeg operatora, koji stvara objekt i vraća poziv na objekt. Objektni tipovi (npr. string, integer) se jako razlikuju od primitivnih tipova (npr. int). Jednostavno deklariranje varijable čiji je tip

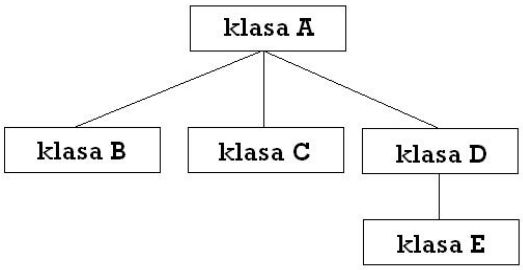
zadan klasom ne znači odmah i kreiranje objekta te klase. Objekti moraju biti izričito konstruirani. Za računalo, postupak konstruiranja objekta znači pronalaženje slobodne memorije za spremanje objekta i punjenje varijabli instance tog objekta. Programera obično ne zanima gdje će u memoriji objekt biti spremljen, ali je poželjno da ima bar nekakav nadzor nad početnim vrijednostima spremljenim u varijable instance objekta. Ako se varijabli instance ne pridijeli početna vrijednost, automatski će joj biti pridijeljena osnovna vrijednost. Varijablama instance numeričkog tipa (int, double, ...) se pridjeljuje početna vrijednost 0, logičkim varijablama (boolean) se pridjeljuje vrijednost false, a char varijablama se pridjeljuje Unicode znak brojčane vrijednosti 0. Varijable instance također mogu biti i tipa objekt. Osnovna početna vrijednost za takve varijable je null, što vrijedi i za string varijable koje su isto tako objekti.

Svaka klasa ima svoj konstruktor, a ako ga programer nije zadao, sistem pridjeljuje osnovni konstruktor. Dakle, konstruktor ne mora biti prikazan u definiciji klase, ali, u svakom slučaju, postoji. Osnovni konstruktor ne radi ništa osim pridjeljivanja memorije i inicijalizacije varijabli instance. Za sve druge potrebe moguće je uključiti jedan ili više konstruktora u definiciju klase. Definicija konstruktora izgleda jednako kao i definicija bilo kojeg drugog potprograma, uz tri iznimke. Konstruktor nema povratnog tipa, ime konstruktora mora biti jednako kao i ime klase u kojoj je definiran, a jedini modifikatori koji mogu biti korišteni na konstruktoru su modifikatori pristupa public, private i protected (konstruktor ne može biti deklariran static). Konstruktor ima uobičajenu građu potprograma, tj. blok naredbi. Nema ograničenja naredbi koje mogu biti korištene. Jedan od glavnih razloga korištenja konstruktora je mogućnost uključivanja parametara koji daju podatke korištene pri kreiranju objekta. Poziv konstruktora je složeniji od običnog poziva potprograma ili funkcije. Koraci kroz koje računalo prolazi pri pozivu konstruktora:

- računalo osigurava dio neiskorištene memorije, dovoljno veliko za spremanje objekta određenog tipa
- inicijalizacija varijabli instance objekta. Ako su zadane početne vrijednosti, te se vrijednosti izračunavaju i spremaju u varijable, u suprotnom se varijablama pridjeljuju osnovne vrijednosti
- procjena vrijednosti stvarnih parametara u konstruktoru, te se pridjeljuju formalnim parametrima konstruktora
- izvršavanje naredbi iz tijela konstruktora, ako ih uopće ima
- vraćanje poziva na objekt kao vrijednost poziva konstruktora

Krajnji rezultat svega ovoga je poziv na novi objekt, koji se može koristiti za pristup varijablama instance ili poziv metoda instance tog objekta.

Klasa predstavlja skup objekata sa zajedničkom građom i ponašanjem. Klasa određuje strukturu objekta navođenjem varijabli koje su sadržane u svim instancama klase i određuje ponašanje objekata preko metode instance koje izražavaju ponašanje objekata. Ovo je moćna ideja, ali nešto poput ovog se može postići u većini programskih jezika. Glavna novost objektno orijentiranog programiranja u odnosu na tradicionalno je da klase mogu izražavati sličnosti između objekata koji imaju zajedničke neke, ali ne sve dijelove strukture i ponašanja. Ovakve sličnosti mogu biti izražene pomoću nasljeđivanja. Naziv nasljeđivanje odnosi se na činjenicu da jedna klasa može naslijediti dio ili svu strukturu i ponašanje od druge klase. Klasa koja nasljeđuje zove se podklasa (subclass) klase od koje nasljeđuje. Ako je klasa B podklasa klase A onda ja klasa A nadklasa (superclass) klase B. Podklasa može nadopunjavati strukturu i ponašanje klase koju nasljeđuje, a može i zamijeniti ili izmijeniti naslijeđeno ponašanje, ali ne i naslijeđenu strukturu. Više se klasa može deklarirati kao podklase iste nadklase. Podklase (mogu se zvati i srodne klase) imaju zajedničke dijelove strukture i ponašanja - one koje nasljeđuju od zajedničke nadklase. Nadklasa izražava ove zajedničke strukture i ponašanja. Na slici (Slika 26), klase B, C i D su srodne klase. Nasljeđivanje se može protegnuti preko nekoliko generacija klasa, kao na slici, gdje je klasa E podklasa klase D koja je, opet, podklasa klase A. U ovom slučaju, klasa E smatra se podklasom klase A iako joj nije izravna podklasa.



Slika 26. Klase i podklase

5.2 Programski jezik Delphi7

Slika 27. Delphi7 početna slika

Delphi je objektno orijentiran programski jezik baziran na Object Pascalu. Prema tome, on se bitno razlikuje od samog Pascala koji je klasični programski jezik. Iako imaju istu sintaksu, Delphi se koristi gotovim objektima čije se instance koriste za interakciju sa korisnikom u razvojnoj aplikaciji. Tim gotovim objektima određuju se svojstva (properties) kao što su boja, položaj, natpis i slično, te reakcije na određene događaje (events) kao što su klik mišem, pritisak tipke i slično. Dakle, koristi se kodna sintaksu Pascala za određivanje tih svojstava i reakcije objekata, no ne mora se uistinu potanko kreirati te objekte već oni postoje kao gotovi. Ovakav način rada znatno ubrzava razvoj aplikacija i omogućuje uspješno i brzo kreiranje kvalitetnih aplikacija.

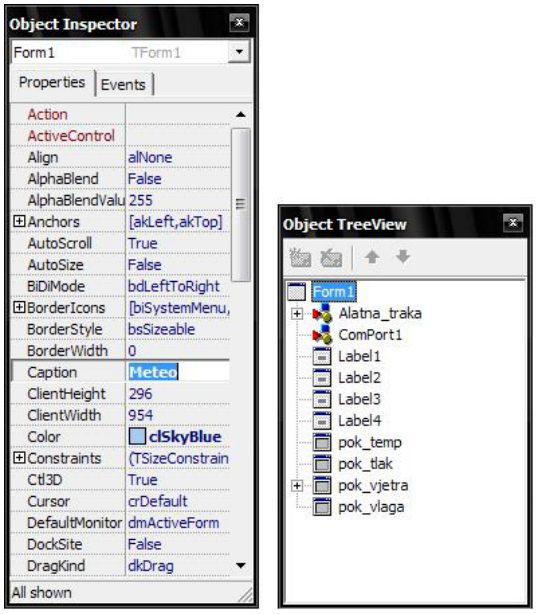
5.2.1 Razvojno okruženje Delphi7

Delphi se sastoji od nekoliko osnovnih prozora koji nam služe za razvoj aplikacija. To su:

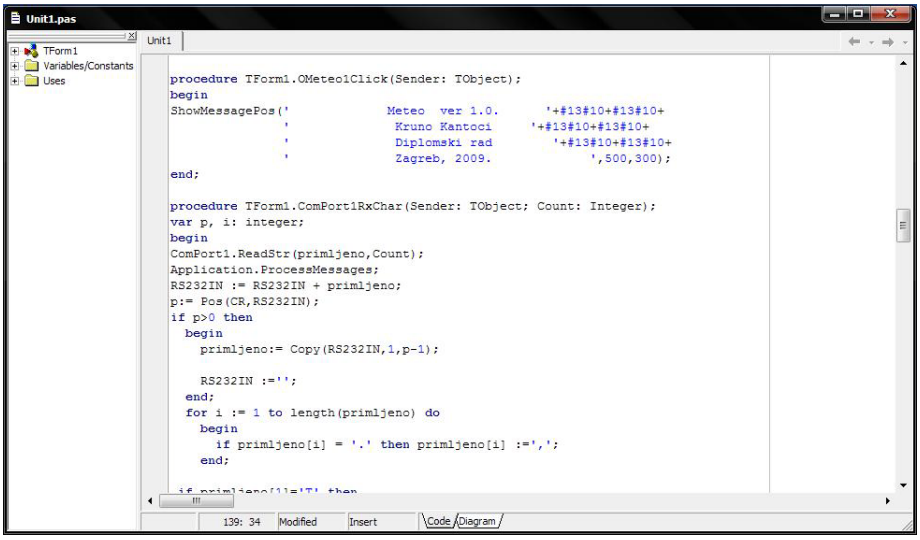
- Speedbar: sadrži nekoliko najvažnijih funkcija kojima možemo brzo pristupiti jednim klikom
- Component Palette: paleta komponenata (vizualnih i nevizualnih) raspoređena u kategorije radi lakšeg snalaženja
- Form window: prozor za uređivanje i pregled forme, tj. aplikacije
- Code window: prozor koji služi za određivanje svojstava i događaja, tj. pisanje programa
- Object Inspector: služi za pregled i promjenu svojstava te nadziranje reakcija na događaje pojedinih objekata na formi
- Object TreeView: pregledni hijerarhijski prikaz svih objekata trenutne forme

Slika 28. Speedbar i Component Palette

Slika 29. Form window



Slika 30. Object Inspector i Object TreeView



Slika 31. Code window

5.2.2 Programiranje u Delphi7

U Delphiju se mogu kreirati popularni „prozori“ (odnosno aplikacije) za Windows i Linux platformu. Da bi kreirali neku aplikaciju, prvo treba odabrati formu. Na tu formu stavljaju se objekti, odnosno instance klasa. Kada je tako objekt određen, treba mu odrediti svojstva i događaje na koje se poziva, odnosno koje operacije utječu na objekt i kako on na njih reagira, odnosno koje operacija sam objekt inicira. Pod svojstvima smatra se njegov položaj na formi, njegovo ime, natpis, boja i slično. Pritom treba paziti na cjelokupni dizajn i vidljivost pojedinih objekata u odnosu na ostale. Koristeći se kodnom sintaksom Pascala, objektu se određuje kako će reagirati u određenoj situaciji i koje potprograme pokreće. Tu je još jedan važan korak kada se razvija tzv. *user-friendly* aplikacija (aplikacija pristupačna korisniku). Naime, treba paziti da program jasno predstavlja svoje funkcije i jasno obavještava korisnika o tijeku procesa i potencijalnim greškama od strane samog korisnika.

Nakon programiranja potrebno je provesti traženje grešaka ili eng. debugging. Naime, program treba podvrgnuti svim mogućim naredbama korisnika i kontrolirati da li se ponaša kako je zamišljeno. Osim toga, treba još jednom provjeriti algoritme i uvjeriti se u njihovu općenitost, odnosno da zadovoljavaju sve moguće postupke od strane korisnika.

Kako se Delphi razvijao kroz godine, tako se on poboljšavao ne samo kao program koji nudi svoje mogućnosti kreiranja aplikacija, već svjestan širine informatičkog svijeta i raznolikosti potreba korisnika, nudi sve veću i lakšu komunikaciju sa ostalim programima. Osnova za povezivanje jedne aplikacije sa drugom je ActiveX tehnologija. Naime, ActiveX je tehnologija koja se koristi određenim datotekama, nazvanim biblioteke (*library*), koje u sebi

sadrže naredbe preko kojih možemo komunicirati a nekom vanjskom aplikacijom. Te datoteke stvara sam program s kojim želimo komunicirati. U tom slučaju, kreirane aplikacije povezuju se sa tom datotekom te postupaju s njom kao s objektom unutar Delphija.

5.3 Izrada računalnog sučelja – Meteo

Osnovni elementi kojima je najbolje pratiti parametre u realnom vremenu su analogni i digitalni pokazivači. Kako ti elementi, tj. njihove biblioteke, nisu standardni elementi Delphija, potrebno ih je ili kreirati ili uvesti već gotove u Delphi razvojno okruženje. Tako je biblioteka koja je naknadno uvezena u Delphi, „TMS Software Components“. Između gomile dodataka koje nudi, nalaze se i analogno – digitalni pokazivači koji nude veliki broj promjenjivih svojstava (properties), tako da je moguće od praktički jednakih komponenata kreirati vizualno potpuno drugačije.

Druga uvezena biblioteka koja je bila potrebna za izradu računalnog sučelja je „CPortLib“. Ona služi za direktno prenošenje podataka iz kreirane računalne aplikacije na serijsku vezu te obrnuto.

5.3.1 Tijek programiranja i korištene funkcije

Na početku, Delphi ispisuje biblioteke koje su potrebne za rad aplikacije. Naime, ako te datoteke nisu učitane, računalo koje nema instaliran Delphi, neće biti u stanju pokrenuti je. Taj dio koda izgreda ovako:

```
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, AdvSmoothGauge, StdCtrls, AdvSmoothButton, AdvSmoothJogWheel,
  AdvSmoothMenu, Menus, CPort, ShellApi;
```

Nakon tipova podataka, ispisane su procedure koje aplikacija koristi. One predstavljaju imena akcija koje su moguće tijekom rada aplikacije:

```
procedure FormCreate(Sender: TObject);
procedure Izlaz1Click(Sender: TObject);
procedure Postavkeveze1Click(Sender: TObject);
procedure OMeteo1Click(Sender: TObject);
procedure ComPort1RxChar(Sender: TObject; Count: Integer);
procedure Uspostavivezu1Click(Sender: TObject);
procedure Prekinivezu1Click(Sender: TObject);
procedure emplog1Click(Sender: TObject);
procedure Vlagalog1Click(Sender: TObject);
procedure laklog1Click(Sender: TObject);
```

Sljedeće je postavljanje varijabli i konstanti. Ovdje će biti samo ispisane, ali ako će se pojaviti potreba opisivanja, u narednom dijelu će biti.

```
const CR=#13;
var
  Form1: TForm1;
  temp, tlak, vlaga, vjetar, smjer : double;
  tempi : integer;
  temps, primljeno, SP_IN, RS232IN, sat, datum  : string;
  temp_log, tlak_log, vlaga_log  : TextFile;
  vrijeme: TDateTime;
```

Da bi prozor aplikacije izgledao na željeni način, potrebno je definirati početne parametre. Većinu tih parametara moguće je podesiti u već spomenutom „Object Inspectoru“, ali neke nije. Ovaj dio koda prikazuje da se pri pokretanju aplikacije otvara serijska komunikacija, podešavaju svojstva menija te postavljaju početne vrijednosti kazaljki tlaka i vlage.

```
pok_tlak.MinimumValue := 900;
uspostavivezu1.Enabled := false;
prekinivezu1.Enabled := true;
pok_vlaga.Value := 0;
```

Nadalje, kako „Meteo“ sprema sve primljene podatke u tekst datoteke, potrebno je provjeriti da li te datoteke postoje, te ako ne postoje, trebaju biti kreirane. Ako kao primjer uzmemo spremanje vrijednosti temperature, pripadajuća datoteka je „temp_log.txt“. Najprije je potrebno provjeriti ako ta datoteka postoji u mapi u kojoj se nalazi Meteo.exe. Ako postoji, onda nije potrebno istu kreirati, a u suprotnom, kreira se na način da se navede kojom datotekom je potrebno manipulirati (naredba *AssignFile*), njezinim kreiranjem (*ReWrite*) te, da bi se moglo pristupiti nekoj drugoj datoteci, njezinim zatvaranjem (*CloseFile*). Prije zatvaranja ovdje je još dodana naredba *WriteLn* kojom se u prvi redak tekst datoteke ispisuju značajke pojedinog stupca koji će kasnije biti kreiran (datum, vrijeme te mjerna jedinica).

```
if FileSearch('temp_log.txt', GetCurrentDir) = ' ' then
  begin
     AssignFile(temp_log, 'temp_log.txt');
     ReWrite(temp_log);
     WriteLn(temp_log,'Datum','     ','Vrijeme','  ','°C' ); WriteLn(temp_log,' ');
     CloseFile(temp_log);
  end;
```

Glavni izbornik nudi mogućnost izlaza iz programa, promjenu postavki serijske komunikacije (za naprednije korisnike), te informaciju o autoru i namjeni programa.

```
procedure TForm1.Izlaz1Click(Sender: TObject);
begin
```

```
Close;
end;

procedure TForm1.Postavkeveze1Click(Sender: TObject);
begin
  comport1.ShowSetupDialog;
end;

procedure TForm1.OMeteo1Click(Sender: TObject);
begin
ShowMessagePos('          Meteo  ver 1.0.     '+#13#10+#13#10+
                '          Kruno Kantoci    '+#13#10+#13#10+
                '          Diplomski rad      '+#13#10+#13#10+
                '          Zagreb, 2009.          ',500,300);
end;
```

Kreirane datoteke koje sadrže spremljene podatke moguće je otvoriti izvan same aplikacije, ali potrebno je korisniku omogućiti da to isto napravi i u samoj aplikaciji. Ako kao primjer uzmemo temperaturu, onda taj dio koda izgleda ovako:

```
procedure TForm1.emplog1Click(Sender: TObject);
begin
ShellExecute(Handle, 'open', 'temp_log.txt', nil, nil, SW_SHOWNORMAL);
end;
```

Komunikacija računala i kontrolera odvija se u jednom smjeru, tj. aplikacija Meteo je izrađena na takav način da samo prikazuje vrijednosti dobivene serijskom vezom. Primanje tih podataka se izvodi na način da se primljeni paket spremi u varijablu *primljeno*, te se ujedno varijablom *Count* broji koliko znakova je primljeno. Nadalje, primljeni paket se zapisuje u novu varijablu *RS232IN* iz koje se očitava na kojem mjestu je znak *CR* koji predstavlja kraj samog paketa (#13). Tada se, bez znaka *CR* dobiven paket ponovo sprema u varijablu *primljeno* te je tako sama komunikacija, te obrada primljenog paketa, završena.

```
ComPort1.ReadStr(primljeno,Count);
Application.ProcessMessages;
RS232IN := RS232IN + primljeno;
p:= Pos(CR,RS232IN);
if p>0 then
  begin
    primljeno:= Copy(RS232IN,1,p-1);
    RS232IN :='';
  end;
```

Prvi znak, tj. slovo u varijabli *primljeno* označava koji je podatak primljen. „*T*“ je za temperaturu, „*P*“ za tlak, „*H*“ za relativnu vlažnost, „*W*“ za brzinu vjetra, te konačno „*D*“ za smjer vjetra. Kako je primljeni paket polje znakova, moguće je na vrlo jednostavan način detektirati pripadnost informacije koja slijedi:

```
primljeno[1]
```

Nakon grananja *if* petlje u jednu od pet mogućih stanja, prvi znak više nije potreban jer se treba očitati samo informacija o vrijednosti trenutnog stanja pojedinog parametra vremena. To micanje prvog znaka izvedeno je funkcijom *Delete*.

Ako kao primjer pristigle informacije uzmemo trenutnu relativnu vlažnost, njen prikaz se vrši kako slijedi:

- brisanje prvog znaka paketa

```
Delete(primljeno, 1, 1);
```

- pretvaranje niza znakova (*string*) u broj s pomičnim zarezom (*float*) zbog toga što TMS komponente prepoznaju samo takve varijable. Taj broj se sprema u varijablu *vlaga*

```
vlaga := strtofloat(primljeno);
```

- limitiranje vrijednosti od 0 do 100% relativne vlažnosti

```
if vlaga < 0 then vlaga := 0;
if vlaga > 100 then vlaga := 100;
```

- postavljanje vidljivosti kazaljke pokazivača (pri prvom prikazivanju), te postavljanje njegove vrijednosti na vrijednost primljenu serijskom vezom

```
pok_vlaga.Value := vlaga;
pok_vlaga.Digit.Visible:= true;
```

- očitavanje trenutnog vremena i datuma te njihovo spremanje u varijable *sat* i *datum*

```
sat := TimeToStr(Time);
datum := DateToStr(Date);
```

- postavljanje znaka pomoći (eng. hint) koji se ispisuje pri prelasku mišem preko pokazivača relativne vlažnosti. Njime se prikazuje vrijeme zadnje promjene stanja pokazivača

```
pok_vlaga.Hint := 'Zadnja promjena ' + sat;
```

- otvaranje tekstualne datoteke *„vlaga_log.txt“* te upisivanje datuma, vremena, te vrijednosti relativne vlažnosti. Zatvaranje datoteke.

```
AssignFile(vlaga_log, 'vlaga_log.txt');
Append(vlaga_log);
WriteLn(vlaga_log, datum,'  ',sat,'  ',FloatToStr(vlaga));
CloseFile(vlaga_log);
```

Na istom principu se vrše promjene svih ostalih indikatora stanja u računalnoj aplikaciji Meteo. Cijeli kod programa moguće je vidjeti u prilogu (Poglavlje 10.1)

5.4 Programiranje mikrokontrolera

Programiranje mikrokontrolera izvedeno je pomoću *Keil Microvision* programskog paketa, koji koristi kodnu sintaksu C-a. Prednost Microvisiona u odnosu na klasično programiranje je ta što, nakon odabira mikrokontrolerske platforme, on brine o samoj arhitekturi kontrolera, tako da programer ne mora misliti o svim detaljima. Nakon samog programiranja, samo pritiskom na gumb moguće je izvršiti kompajliranje, čime se dobiva *.hex* datoteka koju je na posljetku samo potrebno „ubaciti“ u mikrokontroler. Taj prijenos *.hex* datoteke izveden je pomoću računalne aplikacije *Flip.* Zbog toga što su i Flip, a i mikrokontroler proizvodi tvrtke *Atmel*, samo prebacivanje je kompatibilno sa svim Atmel mikrokontrolerima.

5.4.1 Tijek programiranja i korištene funkcije

Cijeli program za upravljanje mikrokontrolerom je izveden korištenjem podprograma koji se po potrebi pozivaju u glavnom programu. Tako je glavni program vrlo jednostavan. Za podatak o određenom parametru vremena najprije se poziva podprogram za mjerenje istog, te nakon toga podprogram za njegovo prikazivanje na LCD ekranu i slanje serijskom vezom prema računalu.

Na početku je potrebna inicijalizacija LCD ekrana i serijske veze. Nakon toga se mjere i ispisuju svi parametri vremena, kako slijedi:

```
InitPrikaz();

Mjerenje_Temp_i_Vlage();
PrikazTemp(temp);
Delay(1000);

tlak = MjerenjeTlak();
PrikazTlak(tlak);
Delay(1000);

PrikazVlaga(vlaga);
Delay(1000);

brzina = MjerenjeBrzina();
PrikazBrzina(brzina);
Delay(1000);

smjer =  MjerenjeSmjer();
PrikazSmjer(smjer);
Delay(5000);
```

Nakon toga slijedi sekvenca ponavljanja. Kako je vjetar najdinamičniji parametar vremena, njegovo uzorkovanje se vrši svakih 8 sekundi. Uzorkovanje ostalih parametara izvodi se svaku 1 minutu, tj., nakon 8 uzorkovanja brzine i smjera vjetra.

```
while (1)
        {
                for (iter = 0; iter < 8; iter++)
                {
                        brzina = MjerenjeBrzina();
                        PrikazBrzina(brzina);
                        Delay(1000);

                        smjer =  MjerenjeSmjer();
                        PrikazSmjer(smjer);
                        Delay(5000);
                }
                Mjerenje_Temp_i_Vlage();
                PrikazTemp(temp);
                Delay(1000);

                tlak = MjerenjeTlak();
                PrikazTlak(tlak);
                Delay(1000);

                PrikazVlaga(vlaga);
                Delay(1000);
        };
```

U glavnom programu moguće je vidjeti koji se sve podprogrami izvršavaju. Prikazivanje i slanje prema računalu, za sve se parametre vremena izvodi sličnom sintaksom, a kao primjer će biti navedena temperatura. Varijabla *temp* uvijek dolazi u cjelobrojnom obliku u rasponu od -300 do 500, što predstavlja vrijednosti od -30 do 50 °C. Ako se pak slučajno desi da je manja od -300 onda mora poprimiti vrijednost -300, a ako je veća od 500, poprima vrijednost 500. Tada se dobivena vrijednost dijeli sa 10, pri čemu se cijeli broj dijeljenja zapisuje u varijablu *tempc*, a ostatak u *tempo*. Tako smo naizgled broj sa pomičnim zarezom zapisali u dva cjelobrojna, čime je sačuvan memorijski prostor. Nakon toga se serijskom vezom na računalo šalje znak „T“ kojemu se priljepljuje cjelobrojna vrijednost temperature, znak pomičnog zareza, decimalna vrijednost temperature, te na kraju, znak „#13“ koji označava kraj paketa. Istim se principom ispisuje vrijednost temperature na LCD ekran, s tom razlikom da se ne zapisuje znak „T“, već se mora zadati lokacija prvog znaka na samom LCD ekranu. Opisana sintaksa je sljedeća:

```
void PrikazTemp (signed int t)
{
   data signed int tempo, tempc;
   if (t>500) t = 500;
   if (t<-300) t = -300;
```

```
    tempc = t / 10;
    tempo = abs(t) % 10;
    PrintChar('T'); PrintSInt(tempc,3); PrintChar(','); PrintInt(tempo,1); PrintChar(CR);
    MoveCursor(1,14); WriteString("      "); MoveCursor(1,14);
    WriteSInt(tempc,3); WriteChar(','); WriteInt(tempo,1); WriteString("°C");
}
```

Pri očitavanju stanja sa senzora, za početak je nužno definirati pripadnost pojedinih pinova mikrokontrolera u odnosu na njihovu vezu sa izlaznim pinovima pojedinog senzora. Taj korak moguće je zaobići, ali je kasnije vrlo teško pratiti njihovu vezu.

```
sbit BrSig = P1^5;
sbit stanje1 = P1^1;
sbit stanje2 = P1^2;
sbit stanje4 = P1^3;
sbit stanje8 = P1^4;
sbit CLKP = P3^6;
sbit CSP  = P3^7;
sbit Dout = P3^5;
sbit DATA = P3^4;
sbit SCK = P3^3;
```

Brzina vjetra se mjeri na temelju broja impulsa koje senzor prima u određenoj jedinici vremena. Zbog toga je definirana sljedeća sekvenca koja odbrojava 100 ms:

```
Timer2() interrupt 5
{
        if (Tint > 0) Tint--;
        if (Tint == 0)
        {
                sec++;
                Tint=100;
   };
        TF2 = 0;
}
```

Kao što je vidljivo, definirana je sekvenca *Timera 2*, kojim se povećava vrijednost varijable *sec* za svaku iteraciju. Da bi se izmjerila brzina vjetra, unutar 2 sekunde se broje *BrSig* impulsi. To je izvedeno tako da se odredi vrijeme intervala (*Tint*) od 100 ms, te za početak postavi varijabla *sec* na nulu. Tada se postave parametri *Timera 2* te se isti pokrene. Da bi se *BrSig* postavio kao ulazni pin, potrebno ga je postaviti visoko (u jedinicu). Nakon toga brojač odbrojava 2 sekunde, a za to vrijeme pristigli impulsi pribrajaju se varijabli *BrImp*. Njena vrijednost, kada prođe postavljeno vrijeme brojenja, predstavlja broj impulsa koji su primljeni. Mjerenjem stvarne brzine vjetra u odnosu na broj impulsa, dobiven je koeficijent proporcionalnosti *KoefBr* koji iznosi 0,62. Množenjem varijable *BrImpF* sa njim, dobiva se

prava vrijednosti brzine vjetra. Nakon toga se varijabla *BrImpF* množi sa 10 te joj se pribraja 0,5, čime je omogućen njezin prikaz kao cjelobrojnu vrijednost koja je spremljena u varijabli *BrImpI*.

```
unsigned int MjerenjeBrzina(void)
{
        Tint = 100; sec = 0; nsec = sec; BrImp = 0;
        SetTimer(2,CTimer);
        StartTimer(2);
  BrSig=1;
        while (sec < odbrojavanje)
        {
                if (BrSig == 0)
                {
                  BrImp++;
                  while (BrSig==0);
                };
        };
        StopTimer(2);
        BrImpF = BrImp * KoefBr;
        BrImpF = BrImpF *10;
        BrImpF = BrImpF + 0.5;
        BrImpI= (int)BrImpF;
        return(BrImpI);
}
```

Podprogram za praćenje smjera vjetra je krajnje jednostavan. Kako se kao izlaz senzora dobiva 4 – bitni binarni broj, jedino što je potrebno jest očitati pojedini bit i pridružiti mu odgovarajuću vrijednost decimalnog sustava. Tako je vrijednost najnižeg bita jednaka 2^0, tj. jedinici, a najvišeg 2^3, tj. osmici. Na kraju je samo potrebno osnovnom operacijom zbrajanja dobiti konačnu vrijednost senzora. Ovdje je važno napomenuti da se prije samog očitavanja stanja senzora, sve linije spuštaju nisko, a kada se očitavanje završi, natrag dižu visoko.

```
unsigned int MjerenjeSmjer(void)
{
        data unsigned int stanje, S1, S2, S4, S8;

        P1 = 0xE1;
        if (stanje1 == 1) S1 = 1;
        if (stanje1 == 0) S1 = 0;
        if (stanje2 == 1) S2 = 2;
        if (stanje2 == 0) S2 = 0;
        if (stanje4 == 1) S4 = 4;
        if (stanje4 == 0) S4 = 0;
        if (stanje8 == 1) S8 = 8;
        if (stanje8 == 0) S8 = 0;
        P1 = 0xFF;
        stanje = S1+S2+S4+S8;
        return(stanje);
}
```

Podprogram za mjerenje tlaka sastoji se od dvije cjeline. Kako je prije navedeno, (Poglavlje 4.1.3) senzor tlaka daje analogni signal napona od 0,5 do 4,5V. Analogno – digitalnom pretvorbom, dobivaju se vrijednosti raspona od 0 do 4096. Prvi podprogram, koji očitava stanje A/D pretvornika nazvan je *LTC1298()* i moguće ga je vidjeti u prilogu (Poglavlje 10.2). Drugim podprogramom nazvanim *MjerenjeTlak()*, postavljaju se parametri prijašnjeg, a očitana vrijednost se sprema u lokalnu varijablu *vrijednost*. Naravno, tu vrijednost treba uskladiti sa stvarnim tlakom. Matematički gledano, pretvornik za 0 V daje vrijednost 0, a za 5 V, 4096 . Kako je izlaz senzora minimalno 0,5 V, dobivena vrijednost pretvornika mora biti smanjena za 411,5. To znači da je za tlak od 0 hPa, očitana vrijednost sa pretvornika 411,5. Sa druge strane, maksimalni napon senzora pri tlaku od 1034 hPa (15 psi) je 4,5 V. Preračunato u vrijednost A/D pretvornika, to iznosi 3704. Tako dolazimo do zaključka da je 1 hPa jednak izlaznoj vrijednosti pretvornika od 3,184. Konačan izraz kojim izračunavamo vrijednost tlaka je:

$$tlak = \frac{očitana\ vrijednost - 411{,}5}{3{,}184} + KalibTlak$$

Konstanta *KalibTlak* ovdje predstavlja grešku vakuumske reference senzora tlaka, o kojoj je već bilo riječi (Poglavlje 4.1.3). Kako je varijabla *vrijednost* broj sa pomičnim zarezom, potrebno ju je definirati kao cjelobrojnu. To se radi tako da se ista zbroji sa 0,5 te onda dobiveni broj prebaci u novu varijablu, u ovom slučaju *tlakI*, koja je definirana kao cjelobrojna.

```
unsigned int MjerenjeTlak(void)
{
        data unsigned int vrijednost, tlakI;
        data float tlakF;

        CLKP = 1; CSP = 1; Dout = 1;
        vrijednost = LTC1298(1);
        tlakF = (vrijednost-411.5)/3.184 + KalibTlak;
        tlakF = tlakF + 0.5;
        tlakI= (int)tlakF;
        CLKP = 0; CSP = 0; Dout = 0;
        return(tlakI);
}
```

Sekvenca mjerenja temperature i vlage u većem je dijelu preuzeta sa stranica proizvođača. Podijeljena je na sedam podprograma, koje su okupljene podprogramom *Mjerenje_Temp_i_Vlage()*. Podprogrami prate potrebnu sekvencu koja je opisana u poglavlju 4.1.4. Za te potrebe, definirani su sljedovi za slanje podataka, primanje podataka, početak

komunikacije, reset komunikacije, mjerenje te preračunavanje u konačne vrijednosti. Samo mjerenje izvodi se na sljedeći način:

- definiranje *value* polja koje se sastoji od *humi_val* i *temp_val*

```
value humi_val,temp_val;
```

- resetiranje veze i postavljanje varijable *error* u nulu

```
s_connectionreset();
error=0;
```

- očitavanje vrijednosti temperature i vlage. U ovom koraku potrebno je naglasiti da podprogram kojim se mjere navedeni parametri provjerava prisutnost senzora. Ako su senzori detektirani i dobro rade, oni šalju vrijednosti prema kontroleru. Ako se dogodi da se senzor ne javlja, veza se resetira i pokušava se ponovo. U slučaju da ni nakon 65535 iteracija nije moguće očitati vrijednost sa senzora, i temperatura i vlaga poprimaju vrijednost 0. Nakon dobivanja vrijednosti (ili nule), iste se spremaju kao varijable sa pomičnim zarezom tj. *temp_val.f* odnosno *humi_val.f*.

```
error+=s_measure((unsigned char*) &humi_val.i,&checksum,HUMI);
error+=s_measure((unsigned char*) &temp_val.i,&checksum,TEMP);
if(error!=0) s_connectionreset();
else
{
        humi_val.f=(float)humi_val.i;
        temp_val.f=(float)temp_val.i;
        calc_sth11(&humi_val.f,&temp_val.f);
}
```

Kao što je vidljivo, nakon spremanja varijabli kao brojeve s pomičnim zarezom, poziva se funkcija *calc_sht11()*. Njome se dobivene vrijednosti preračunavaju u stvarne, vodeći se uputom proizvođača. Sintaksa za preračunavanje je sljedeća:

```
void calc_sth11(float *p_humidity ,float *p_temperature)
{
 const float C1=-4.0;
 const float C2=+0.0405;
 const float C3=-0.0000028;
 const float T1=+0.01;
 const float T2=+0.00008;

 float rh=*p_humidity;
```

```
  float t=*p_temperature;
  float rh_lin;
  float rh_true;
  float t_C;

  t_C=t*0.01 - 40;
  rh_lin=C3*rh*rh + C2*rh + C1;
  rh_true=(t_C-25)*(T1+T2*rh)+rh_lin;
  if(rh_true>100)rh_true=100;
  if(rh_true<0.1)rh_true=0.1;

  *p_temperature=t_C;
  *p_humidity=rh_true;

  R_Temp=t_C*10;
  temp=(int)R_Temp;
  R_Humi = rh_true + 0.5;
  vlaga = (int)R_Humi;
}
```

Vidljivo se da su na kraju vrijednosti temperature i vlage spremljene kao cjelobrojne vrijednosti *temp*, odnosno *vlaga*, koje se kasnije prikazuju pozivanjem pripadnih podprograma *PrikazVlaga(vlaga)* i *PrikazTemp(temp)*.

Cijeli kod programa za mikrokontroler moguće je vidjeti u prilogu (Poglavlje 10.2).

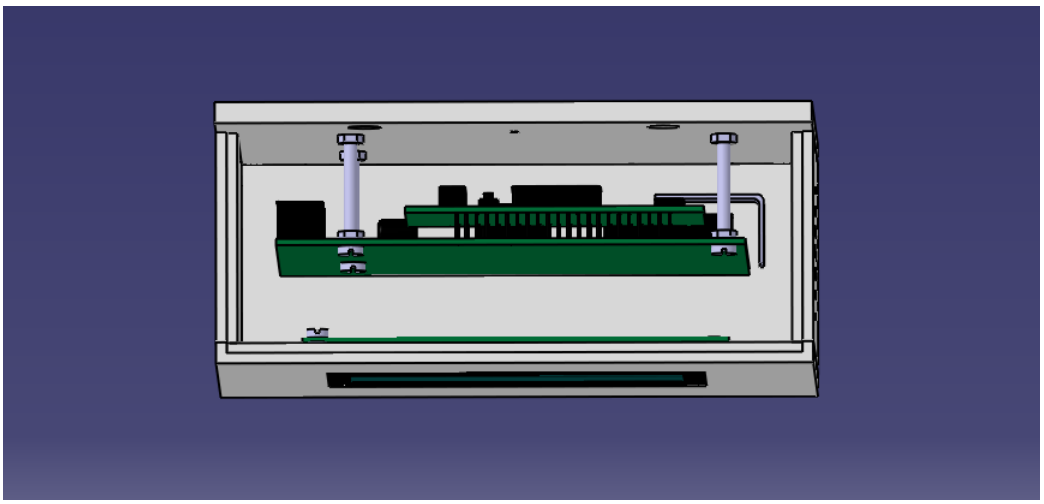
6 MEHANIČKA IZVEDBA

Kao što je ranije spomenuto, mehanička izvedba prototipa meteorološke mjerne stanice sastoji se od dva dijela, unutarnje i vanjske jedinice. Kako mikrokontroler i LCD ekran nisu predviđeni za rad u ekstremnim uvjetima, ta se jedinica stavlja unutra. Time se postiže zaštita od atmosferskih utjecaja, te izbjegava potreba da korisnik mora izaći van da bi provjerio stanje atmosfere. Vanjska jedinica mjeri meteorološke parametre tako da je neizbježna njena vanjska lokacija. Svi elementi zaštićeni su od kiše na način da su stavljeni u vodonepropusnu vanjsku kutiju.

Elektronički dijelovi na slikama modelirani su samo na način da se prikaže raspored elemenata, tako da se vodovi ne vide. Također, nisu prikazana ni ožičenja koja povezuju pojedine elektroničke komponente.

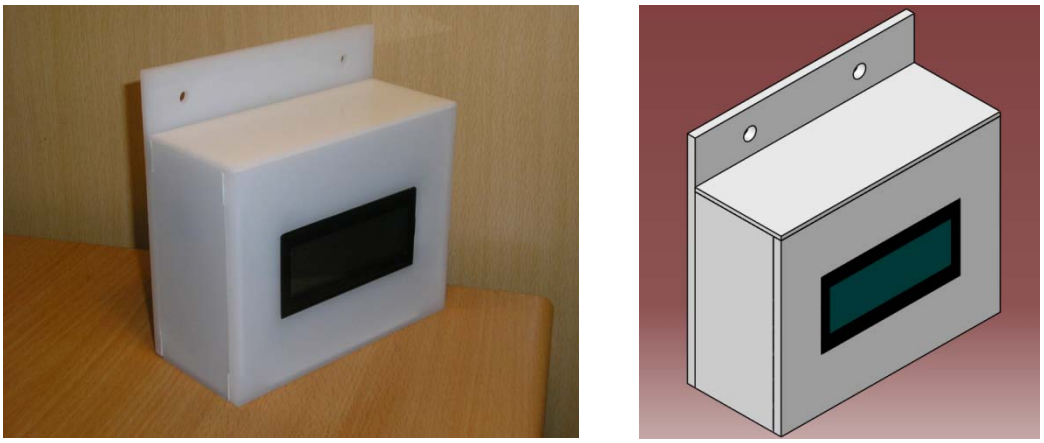
6.1 Unutarnja jedinica

Unutarnja jedinica meteorološke postaje izrađena je od bijelog pleksi – stakla debljine 5 mm. Sastoji se od 6 dijelova koji su zajedno spojeni silikonskim ljepilom. Kako se lijepljeni spojevi ne bi uništili, kutija je izvedena pomoću utora i izdanaka tako da samo kućište preuzima na sebe moguća naprezanja, a lijepljeni spojevi ostaju u prvobitnom stanju. Za pričvršćivanje na zid postoje 2 utora promjera 8 mm sa stražnje strane, dok se svi elektronički spojevi ostvaruju s donje strane kutije. Na donjoj slici (Slika 32), prikazan je raspored elemenata u unutarnjoj jedinici.



Slika 32. Raspored elemenata unutarnje jedinice

Na sljedećoj slici prikazane su fotografija (Slika 33a) i CAD model (Slika 33b) unutrašnje jedinice.

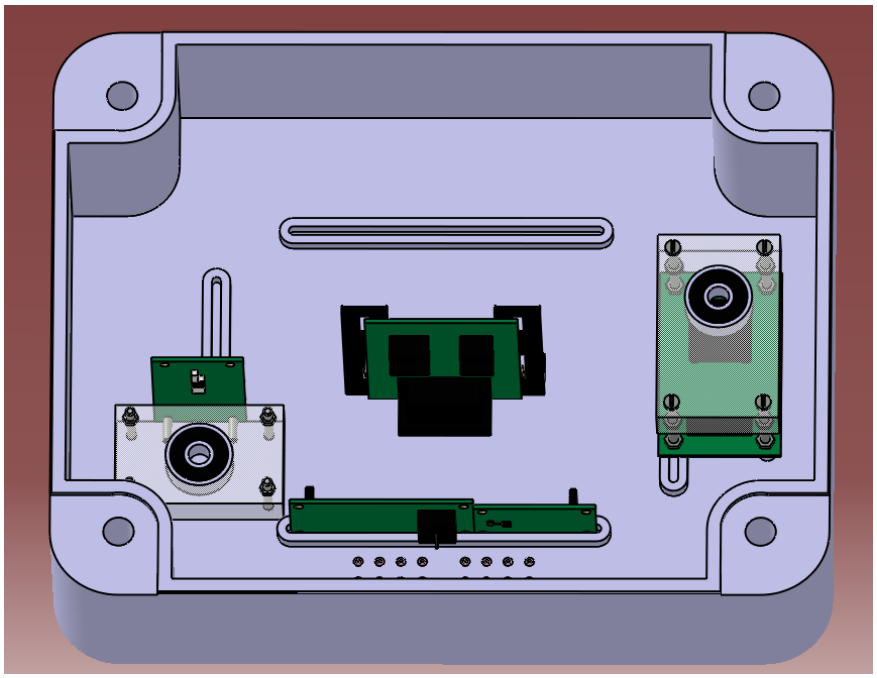


a) b)

Slika 33. Unutarnja jedinica: a) fotografija, b) CAD model

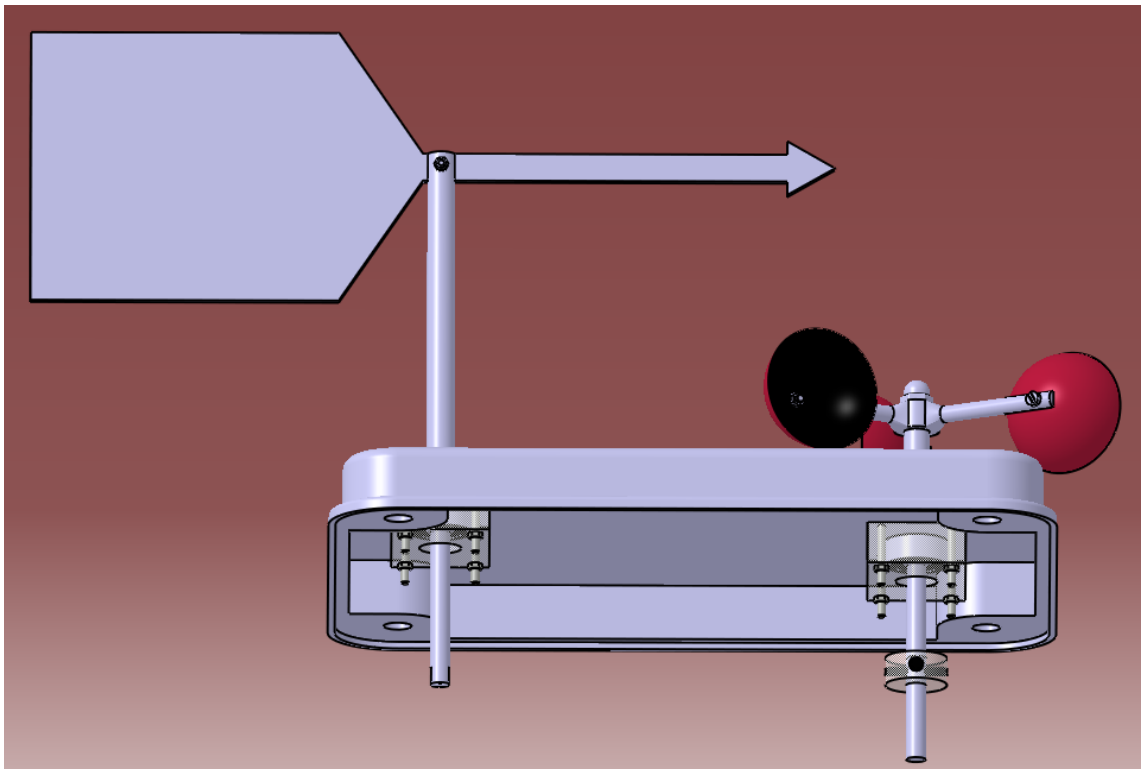
6.2 Vanjska jedinica

Kutija vanjske jedinice je standardna nadžbukna električna kutija veličine 250x195x100 mm. Na donji dio kutije pričvršćeni su senzori za tlak, smjer vjetra, temperaturu i vlagu. Također, na nju je pričvršćen i ekspander senzorskih krugova, za koji je napravljena rupa dimenzija 30x13 mm. Nadalje, na nju su pomoću M3 vijaka učvršćeni držači za ležajeve. Raspored elemenata vidljiv je na donjoj slici (Slika 34).



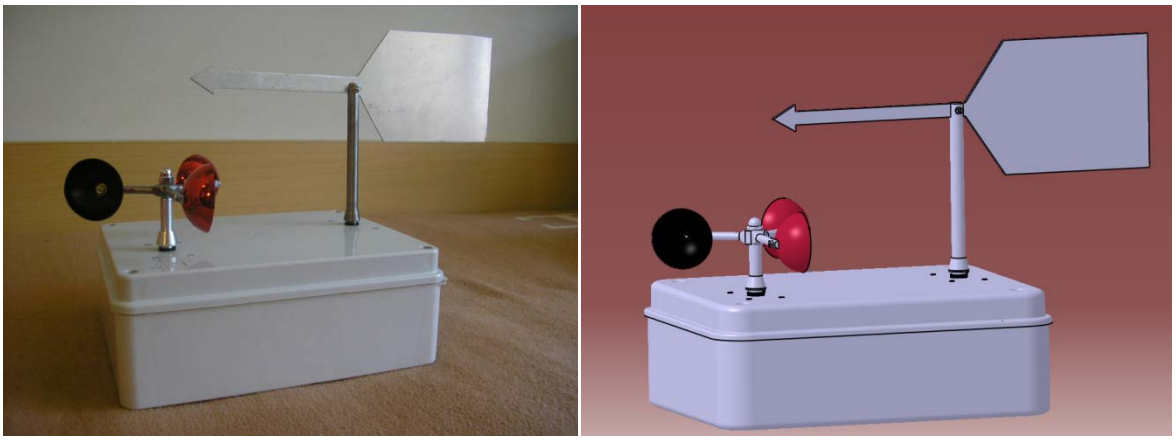
Slika 34. Vanjska jedinica – raspored elemenata na donjem dijelu

Na poklopac vanjske jedinice učvršćeni su držači za radijalno – aksijalne ležajeve. Također, za prolaz osovina izbušeni su provrti promjera 10 mm. Kako ne bi došlo do ulaska vode u kutiju, oko provrta su postavljena dva gumena prstena debljine 2 mm i promjera 14 mm, a kako se ne bi desilo da se voda slijeva sa osovina u unutrašnjost kutije, na njima su postavljena proširenja promjera 16 mm koja vodu odvode izvan gumenih prstena. Na osovinu za mjerenje smjera vjetra stavlja se vjetrulja koja se pričvršćuje M3 vijkom. Osovina za brzinu vjetra na svojem gornjem dijelu ima M6 navoj na koji se pričvršćuje držač čašica i ukrasna matica. Same čašice izvedene su od plastičnih kuglica za bor promjera 40 mm, a pričvršćene su M3 vijcima na držač. Za indikaciju brzine vjetra pomoću senzora, na donji dio osovine postavljen je držač neodimijskog magneta od prozirnog pleksi – stakla. Opisane elemente moguće je vidjeti na donjoj slici (Slika 35).



Slika 35. Vanjska jedinica – raspored elemenata na gornjem dijelu

Na sljedećoj slici prikazane su fotografija (Slika 36a) i CAD model (Slika 36b) vanjske jedinice meteorološke postaje.



a) b)

Slika 36. Vanjska jedinica: a) fotografija, b) CAD model

7 PROCJENA VRIJEDNOSTI INVESTICIJE

Sve navedene cijene pojedinih stavki (komponenata, dijelova i sl.) prototipa meteorološke postaje, koje su navedene u donjoj tablici (Tablica 11) su približne. Također, ovdje neće biti navedeni troškovi prijevoza i telefonskih razgovora koji iznose između 10 do 15% vrijednosti cijele investicije.

Tablica 11. Cijena stavki prototipa meteorološke postaje

Red. br.	Stavka	Količina	Cijena [kn]
1	Sitne elektroničke komponente (konektori, otpornici i sl.)	-	250
2	Mikrokontroler AT89C51ID2	1	50
3	LDC ekran 20x4	1	125
4	Senzor temperature i vlage	1	220
5	Senzor tlaka	1	210
6	Apsolutni enkoder	1	80
7	Euro pločica dvoslojna 100x160mm foto premaz	2	50
8	Adapter 220ACV/9DCV	1	90
9	Nadžbukna kutija	1	90
10	Čelična šipka Ø15 mm	0,5 m	10
11	Aluminijska šipka Ø15 mm	0,5 m	15
12	Ležaj FAG 627 2RS	4	90
13	Razni vijci i spojni elementi	-	50
14	Razne veličine pleksi – stakla	-	100
15	Usluge tokarenja i glodanja	-	250
		UKUPNO	1680

Kao što je vidljivo, cijena prototipa popela se do respektabilnih 1680kn. Ako pretpostavimo da bi takav proizvod krenuo u serijsku proizvodnju, cijena proizvoda bi se spustila za 40–ak posto. Taj pad cijene bio bi uvjetovan količinskim popustom pri kupnji velike serije senzora, manjim troškovima dostave i CNC obradom. Takvim pristupom, cijena jedne meteorološke postaje iznosila bi oko 1000 kn.

8 ZAKLJUČAK

Iako izrada meteorološke mjerne postaje na početku nije djelovala komplicirano, pojavili su se problemi koji nisu bili predviđeni. Neki od njih su pogrešno dovedeni vodovi na pinove mikrokontrolera, nemogućnost izvedbe nekih zamišljenih softverskih rješenja, mehanički problemi prilikom sastavljanja, te mnogi drugi. Njihovo rješavanje bilo je dugotrajno i ponekad zbunjujuće, ali kako sam tokom studiranja naučio razmišljati na način da ništa nije nemoguće, znao sam da za svaki problem postoji rješenje.

Jedan od problema je i dalje ostao neriješen, a tiče se samog mehaničkog apsolutnog enkodera. Kako su vrijeme, a naročito buđet predviđen za izradu ovog rada uvelike premašeni, ovdje će problem biti samo opisan, uz prijedlog novog rješenja. Mehanički enkoder pri svakoj promjeni stanja ostvaruje mehanički kontakt, te se pri tome javlja relativno veliki otpor. To rezultira da se indikator smjera vjetra ne može zakrenut pri brzinama vjetra manjim od 5 m/s. Jedno od rješenja bi bilo da se postavi veća vjetrulja, koja opet ima kao nuspojavu povećanje mase, što nije poželjno. Drugo rješenje bi bilo da se postavi inkrementalni enkoder koji daje impuls pri svakoj promjeni pozicije, ali problem bi se pojavio pri gubitku napajanja senzora, pri čemu bi on izgubio svoju inicijaliziranu vrijednost. Najbolje rješenje je postavljanje optičkog apsolutnog enkodera koji radi na principu prekida svjetlosnog snopa, te time nema mehaničkog otpora. Jedan od takvih je senzor proizvođača Grayhill oznake 63A256-L-B-A.

Kako razvijeni prototip meteorološke postaje nudi mogućnost povezivanja sa vanjskim uređajima, sljedeći korak pri iskorištavanju njegovih mogućnosti je izrada sklopovlja za upravljanje „pametnom kućom“. Poznavanjem pozicija pojedinih prozorskih jedinica kuće, moguće je razviti takav sustav koji će upravljati spuštanjem roleta, čime se dobivaju optimalni uvjeti za boravak. Pri jakom vjetru, dolazi do prijetnje uništavanja prozora, tendi te slične imovine, tako da se pravovremenim reagiranjem opasnost može umanjiti. Kako ljudske aktivnosti uvelike ovise o atmosferskim utjecajima, iskoristivost ovog uređaja je velika.

Sposobnost izrade funkcionalnog uređaja na kraju studiranja ulijeva mi nadu da ću se moći nositi sa svim problemima koji će pred mene biti stavljeni u budućnosti. Iako sam studiranjem naučio principe konstruiranja, elektrotehnike, elektronike, automatike i svih ostalih grana koje su mi bile ponuđene na ovom smjeru, svjestan sam da mehatronika kao grana strojarstva još u potpunosti nije shvaćena, a kamoli istražena i primijenjena. Zbog toga znam da me čeka još mnogo učenja i usavršavanja. Ali, spreman sam na to jer uspješan strojar uči strojarstvo i razmišlja „strojarski“ sve dok je živ.

9 LITERATURA

[1] Penzar B. i suradnici: Meteorologija za korisnike, Školska knjiga, Zagreb, 1996.

[2] Upute proizvođača senzora SHT71 – dostupno na http://www.sensirion.ch/

[3] Upute proizvođača senzora ASDX015A24R – dostupno na http://www.sensortechnics.com/

[4] Upute proizvođača senzora 25LB22-H – dostupno na http://www.grayhill.com/

[5] Karakteristike meteorološke postaje MK-III-RTN – dostupno na www.rainwise.com/

[6] Karakteristike meteorološke postaje Vantage Pro2 – dostupno na www.davisnet.com/

[7] Karakteristike meteorološke postaje WMR 200 – dostupno na www.oregonscientific.com/

[8] Crneković M.: Materijali za predavanja i vježbe iz mikroprocesorskog upravljanja, 2009. – dostupno na http://karmela.fsb.hr/kontroleri/

[9] http://www.delphibasics.co.uk/

10 PRILOG

10.1 Programski kod računalne aplikacije – Meteo

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, AdvSmoothGauge, StdCtrls, AdvSmoothButton, AdvSmoothJogWheel,
  AdvSmoothMenu, Menus, CPort, ShellApi;

type
  TForm1 = class(TForm)
    pok_temp: TAdvSmoothGauge;
    pok_tlak: TAdvSmoothGauge;
    Label1: TLabel;
    Label2: TLabel;
    pok_vlaga: TAdvSmoothGauge;
    Label3: TLabel;
    pok_vjetra: TAdvSmoothGauge;
    oznaka_smjer: TLabel;
    Label4: TLabel;
    Alatna_traka: TMainMenu;
    ComPort1: TComPort;
    Datoteka1: TMenuItem;
    Postavke1: TMenuItem;
    Pomo1: TMenuItem;
    Izlaz1: TMenuItem;
    Postavkeveze1: TMenuItem;
    OMeteo1: TMenuItem;
    Uspostavivezu1: TMenuItem;
    Prekinivezu1: TMenuItem;
    emplog1: TMenuItem;
    Vlagalog1: TMenuItem;
    laklog1: TMenuItem;
    procedure FormCreate(Sender: TObject);
    procedure Izlaz1Click(Sender: TObject);
    procedure Postavkeveze1Click(Sender: TObject);
    procedure OMeteo1Click(Sender: TObject);
    procedure ComPort1RxChar(Sender: TObject; Count: Integer);
    procedure Uspostavivezu1Click(Sender: TObject);
    procedure Prekinivezu1Click(Sender: TObject);
    procedure emplog1Click(Sender: TObject);
    procedure Vlagalog1Click(Sender: TObject);
    procedure laklog1Click(Sender: TObject);
  end;
```

```
  const CR=#13;
  var
  Form1: TForm1;
  temp, tlak, vlaga, vjetar, smjer : double;
  tempi : integer;
  temps, primljeno, SP_IN, RS232IN, sat, datum  : string;
  temp_log, tlak_log, vlaga_log  : TextFile;
  vrijeme: TDateTime;

implementation

{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin
pok_tlak.MinimumValue := 900;
uspostavivezu1.Enabled := false;
prekinivezu1.Enabled := true;
pok_vlaga.Value := 0;

if FileSearch('temp_log.txt', GetCurrentDir) = '' then
  begin
     AssignFile(temp_log, 'temp_log.txt');
     ReWrite(temp_log);
     WriteLn(temp_log,'Datum','     ','Vrijeme','  ','°C' ); WriteLn(temp_log,' ');
     CloseFile(temp_log);
  end;

if FileSearch('tlak_log.txt', GetCurrentDir) = '' then
  begin
     AssignFile(tlak_log, 'tlak_log.txt');
     ReWrite(tlak_log);
     WriteLn(tlak_log,'Datum','     ','Vrijeme','  ','hPa' ); WriteLn(tlak_log,' ');
     CloseFile(tlak_log);
  end;


if FileSearch('vlaga_log.txt', GetCurrentDir) = '' then
  begin
     AssignFile(vlaga_log, 'vlaga_log.txt');
     ReWrite(vlaga_log);
     WriteLn(vlaga_log,'Datum','     ',' Vrijeme','  ','%' ); WriteLn(vlaga_log,' ');
     CloseFile(vlaga_log);
  end;
end;

procedure TForm1.Izlaz1Click(Sender: TObject);
begin
          Close;
end;
```

```
procedure TForm1.Postavkeveze1Click(Sender: TObject);
begin
  comport1.ShowSetupDialog;
end;

procedure TForm1.OMeteo1Click(Sender: TObject);
begin
ShowMessagePos('          Meteo  ver 1.0.      '+#13#10+#13#10+
               '          Kruno Kantoci     '+#13#10+#13#10+
               '          Diplomski rad       '+#13#10+#13#10+
               '          Zagreb, 2009.          ',500,300);
end;

procedure TForm1.ComPort1RxChar(Sender: TObject; Count: Integer);
var p, i: integer;
begin
ComPort1.ReadStr(primljeno,Count);
Application.ProcessMessages;
RS232IN := RS232IN + primljeno;
p:= Pos(CR,RS232IN);
if p>0 then
 begin
  primljeno:= Copy(RS232IN,1,p-1);
  RS232IN :='';
 end;
 for i := 1 to length(primljeno) do
  begin
   if primljeno[i] = '.' then primljeno[i] :=',';
  end;

 if primljeno[1]='T' then
  begin
   Application.ProcessMessages;
   Delete(primljeno, 1, 1);
   temp := strtofloat(primljeno);
    if temp < -30 then temp := -30;
    if temp > 50 then temp := 50;
   pok_temp.Digit.Visible:= true;
   pok_temp.Value := temp;
    if ((temp > -30) and (temp <= -20)) then pok_temp.InnerCircle.gloss.Color := clWhite;
    if ((temp > -20) and (temp <= -5)) then pok_temp.InnerCircle.gloss.Color :=
RGB(110,180,255);
    if ((temp >-5)  and (temp <=10)) then pok_temp.InnerCircle.gloss.Color :=
RGB(80,150,90);
    if ((temp >10)  and (temp <=25)) then pok_temp.InnerCircle.gloss.Color :=
RGB(130,255,100);
    if ((temp >25)  and (temp <=35)) then pok_temp.InnerCircle.gloss.Color :=
RGB(255,125,0);
    if ((temp >32)  and (temp <=50)) then pok_temp.InnerCircle.gloss.Color := clred;
```

```
      sat := TimeToStr(Time);
      datum := DateToStr(Date);
      pok_temp.Hint := 'Zadnja promjena '+ sat;
     AssignFile(temp_log, 'temp_log.txt');
     Append(temp_log);
     WriteLn(temp_log, datum,' ',sat,' ',FloatToStr(temp));
     CloseFile(temp_log);
   end;

 if primljeno[1]='P' then
   begin
     Application.ProcessMessages;
     Delete(primljeno, 1, 1);
     tlak := strtofloat (primljeno);
     if tlak < 900 then tlak := 900;
     if tlak > 1050 then tlak := 1050;
     pok_tlak.Value := tlak;
     pok_tlak.Digit.Visible:= true;
     sat := TimeToStr(Time);
     datum := DateToStr(Date);
     pok_tlak.Hint := 'Zadnja promjena '+ sat;
     AssignFile(tlak_log, 'tlak_log.txt');
     Append(tlak_log);
     WriteLn(tlak_log, datum,' ',sat,' ',FloatToStr(tlak));
     CloseFile(tlak_log);
   end;

 if primljeno[1]='H' then
   begin
     Application.ProcessMessages;
     Delete(primljeno, 1, 1);
     vlaga := strtofloat(primljeno);
       if vlaga < 0 then vlaga := 0;
       if vlaga > 100 then vlaga := 100;
     pok_vlaga.Value := vlaga;
     pok_vlaga.Digit.Visible:= true;
     sat := TimeToStr(Time);
     datum := DateToStr(Date);
     pok_vlaga.Hint := 'Zadnja promjena '+ sat;
     AssignFile(vlaga_log, 'vlaga_log.txt');
     Append(vlaga_log);
     WriteLn(vlaga_log, datum,'  ',sat,' ',FloatToStr(vlaga));
     CloseFile(vlaga_log);
   end;

 if primljeno[1]='W' then
   begin
     Application.ProcessMessages;
     Delete(primljeno, 1, 1);
     vjetar := strtofloat(primljeno);
```

```
      if vjetar < 0 then vjetar := 0;
      if vjetar > 30 then vjetar := 30;
    pok_vjetra.Value := vjetar;
    pok_vjetra.Digit.Visible:= true;
    pok_vjetra.Hint := 'Zadnja promjena '+ (timetostr(now));
  end;

 if primljeno[1]='D' then
  begin
    Application.ProcessMessages;
    Delete(primljeno, 1, 1);
    if primljeno = 'S' then
      begin
        oznaka_smjer.Left := 107;
        oznaka_smjer.Caption :='S';
      end;
   if primljeno = 'Z' then
      begin
        oznaka_smjer.Left := 107;
        oznaka_smjer.Caption :='Z';
      end;
   if primljeno = 'I' then
      begin
        oznaka_smjer.Left := 109;
        oznaka_smjer.Caption :='I';
      end;
   if primljeno = 'J' then
      begin
        oznaka_smjer.Left := 107;
        oznaka_smjer.Caption :='J';
      end;
   if primljeno = 'SZ' then
      begin
        oznaka_smjer.Left := 99;
        oznaka_smjer.Caption :='SZ';
      end;
   if primljeno = 'JZ' then
      begin
        oznaka_smjer.Left := 99;
        oznaka_smjer.Caption :='JZ';
      end;
   if primljeno = 'SI' then
      begin
        oznaka_smjer.Left := 100;
        oznaka_smjer.Caption :='SI';
      end;
   if primljeno = 'JI' then
      begin
        oznaka_smjer.Left := 100;
        oznaka_smjer.Caption :='JI';
```

```
      end;
   if primljeno = 'O' then
      begin
        oznaka_smjer.Left := 100;
        oznaka_smjer.Caption :='  ';
      end;

   end;
end;

procedure TForm1.Uspostavivezu1Click(Sender: TObject);
begin
 comport1.Connected := true;
 prekinivezu1.Enabled := true;
 uspostavivezu1.Enabled := false;
end;

procedure TForm1.Prekinivezu1Click(Sender: TObject);
begin
comport1.Connected := false;
prekinivezu1.Enabled:= false;
uspostavivezu1.Enabled := true;
end;

procedure TForm1.emplog1Click(Sender: TObject);
begin
ShellExecute(Handle, 'open', 'temp_log.txt', nil, nil, SW_SHOWNORMAL);
end;

procedure TForm1.Vlagalog1Click(Sender: TObject);
begin
ShellExecute(Handle, 'open', 'vlaga_log.txt', nil, nil, SW_SHOWNORMAL);
end;

procedure TForm1.laklog1Click(Sender: TObject);
begin
ShellExecute(Handle, 'open', 'tlak_log.txt', nil, nil, SW_SHOWNORMAL);
end;

end.
```

10.2 Programski kod mikrokontrolera

```
/*
-----------------------------------------------------------------
Program:   main.c
Purpose:   Meteoroloska postaja
Revision Date:
Author:   Kruno Kantoci
-----------------------------------------------------------------
*/
#include <c:\Keil7\REG51ID2.h>
#include <c:\Keil7\ID2.h>
#include <string.h>
#include <math.h>
#include <intrins.h>
//-----------------------------------------------------------------

typedef union
{ unsigned int i;
  float f;
} value;
//-----------------------------------------------------------------

#define CR 13
#define odbrojavanje 2
#define CTimer (0xffff-9216)
#define KalibTlak 10

#define noACK 0
#define ACK   1

#define KoefBr 0.62

#define STATUS_REG_W 0x06   //000   0011    0
#define STATUS_REG_R 0x07   //000   0011    1
#define MEASURE_TEMP 0x03   //000   0001    1
#define MEASURE_HUMI 0x05   //000   0010    1
#define RESET        0x1e   //000   1111    0

enum {TEMP,HUMI};
//-----------------------------------------------------------------

sbit BrSig = P1^5;

sbit stanje1 = P1^1;
sbit stanje2 = P1^2;
sbit stanje4 = P1^3;
sbit stanje8 = P1^4;

sbit CLKP = P3^6;
```

```
sbit CSP  = P3^7;
sbit Dout = P3^5;

sbit     DATA = P3^4;
sbit     SCK = P3^3;
//-------------------------------------------------------------

data unsigned int BrImp, Tint, sec, nsec, t, iter;
data unsigned int vlaga, vjetar, tlak, kasnjenje, smjer, brzina;
data signed int temp;
xdata unsigned char RxSTR[10];
xdata float R_Humi, R_Temp;

//-------------------------------------------------------------

Timer2() interrupt 5
{
        if (Tint > 0) Tint--;
        if (Tint == 0)
        {
                sec++;
                Tint=100;
   };
        TF2 = 0;
}
//-------------------------------------------------------------
//---------------temeratura i vlaga-----------------------------

char s_write_byte(unsigned char value)
{
  unsigned char i,error=0;

  for (i=0x80;i>0;i/=2)
  { if (i & value) DATA=1;
    else DATA=0;
    SCK=1;
    _nop_();_nop_();_nop_();
    SCK=0;
  }
  DATA=1;
  SCK=1;
  error=DATA;
  SCK=0;
  return error;
}
//-------------------------------------------------------------

char s_read_byte(unsigned char ack)
{
  unsigned char i,val=0;
```

```
  DATA=1;
  for (i=0x80;i>0;i/=2)
  { SCK=1;
    if (DATA) val=(val | i);
    SCK=0;
  }
  DATA=!ack;
  SCK=1;
  _nop_();_nop_();_nop_();
  SCK=0;
  DATA=1;
  return val;
}
//-----------------------------------------------------------

void s_transstart(void)
{
   DATA=1; SCK=0;
   _nop_();
   SCK=1;
   _nop_();
   DATA=0;
   _nop_();
   SCK=0;
   _nop_();_nop_();_nop_();
   SCK=1;
   _nop_();
   DATA=1;
   _nop_();
   SCK=0;
}
//-----------------------------------------------------------

void s_connectionreset(void)
{
  unsigned char i;

  DATA=1; SCK=0;
  for(i=0;i<9;i++)
  { SCK=1;
    SCK=0;
  }
  s_transstart();
}
//-----------------------------------------------------------

char s_measure(unsigned char *p_value, unsigned char *p_checksum, unsigned char mode)
{
  unsigned error=0;
```

```
  unsigned int i;

  s_transstart();
  switch(mode){
    case TEMP : error+=s_write_byte(MEASURE_TEMP); break;
    case HUMI : error+=s_write_byte(MEASURE_HUMI); break;
    default       : break;
  }
  for (i=0;i<65535;i++) if(DATA==0) break;
  if(DATA) error+=1;
  *(p_value)  =s_read_byte(ACK);
  *(p_value+1)=s_read_byte(ACK);
  *p_checksum =s_read_byte(noACK);
  return error;
}
//----------------------------------------------------------

void calc_sth11(float *p_humidity ,float *p_temperature)
{
  const float C1=-4.0;
  const float C2=+0.0405;
  const float C3=-0.0000028;
  const float T1=+0.01;
  const float T2=+0.00008;

  float rh=*p_humidity;
  float t=*p_temperature;
  float rh_lin;
  float rh_true;
  float t_C;

  t_C=t*0.01 - 40;
  rh_lin=C3*rh*rh + C2*rh + C1;
  rh_true=(t_C-25)*(T1+T2*rh)+rh_lin;
  if(rh_true>100)rh_true=100;
  if(rh_true<0.1)rh_true=0.1;

  *p_temperature=t_C;
  *p_humidity=rh_true;

  R_Temp=t_C*10;
  temp=(int)R_Temp;
  R_Humi = rh_true + 0.5;
  vlaga = (int)R_Humi;
}
//----------------------------------------------------------

void Mjerenje_Temp_i_Vlage(void)
{
```

```
        value humi_val,temp_val;

        unsigned char error,checksum;

        s_connectionreset();

        error=0;
  error+=s_measure((unsigned char*) &humi_val.i,&checksum,HUMI);
  error+=s_measure((unsigned char*) &temp_val.i,&checksum,TEMP);
  if(error!=0) s_connectionreset();
   else
   {
        humi_val.f=(float)humi_val.i;
        temp_val.f=(float)temp_val.i;
        calc_sth11(&humi_val.f,&temp_val.f);

   }
   Delay(1000);

}
//---------------------------------------------------------------
//---------------------------------------------------------------

unsigned int LTC1298(unsigned char Channel)
{
data unsigned char MODE, i, MSB, LSB;
data unsigned int  x;


    CLKP=0; CSP=1;  CSP=0;
    if (Channel > 1) return(65535);
    if (Channel == 0) MODE=0xD0; else MODE=0xF0;

    for (i=0; i<4; i++) {
      if ((MODE & 0x80) == 0) Dout=0; else Dout=1;
      MODE= MODE << 1;
      CLKP=1; CLKP=0;
      };

    MSB=0;
    for (i=0; i<4; i++) {
       MSB= MSB << 1;
       CLKP=1; CLKP=0;
       if (Dout) MSB=MSB | 0x01;
      };
    LSB=0;
    for (i=0; i<8; i++) {
       LSB= LSB << 1;
       CLKP=1; CLKP=0;
       if (Dout) LSB=LSB | 0x01;
```

```
      };
    CSP=1;
    x=256*(int)MSB + LSB;

          return(x);
}

//-------------------------------------------------------------

void PrikazTemp (signed int t)
{
        data signed int tempo, tempc;
        if (t>500) t = 500;
   if (t<-300) t = -300;
   tempc = t / 10;
   tempo = abs(t) % 10;
   PrintChar('T'); PrintSInt(tempc,3); PrintChar(','); PrintInt(tempo,1); PrintChar(CR);
   MoveCursor(1,14); WriteString("       "); MoveCursor(1,14);
   WriteSInt(tempc,3); WriteChar(','); WriteInt(tempo,1); WriteString("°C");
}
//-------------------------------------------------------------

void PrikazBrzina (unsigned int b)
{
        data unsigned int brzc, brzo;

        if (b>300) b= 300;
        brzc = b / 10;
        brzo = abs(b) % 10;
        PrintChar('W'); PrintInt(brzc,2); PrintChar(','); PrintInt(brzo,1); PrintChar(CR);
        MoveCursor(4,9); WriteString("         "); MoveCursor(4,9);
        WriteInt(brzc,2); WriteChar(','); WriteInt(brzo,1); WriteString("m/s");
}
//-------------------------------------------------------------

void PrikazSmjer(unsigned int s)
{
        MoveCursor(4,19); WriteString("  ");

        if (brzina == 0)
        {
        MoveCursor(4,19); WriteString("  "); PrintString("DO"); PrintChar(CR);
   }
        else
        {
                if (s==0)
                {
                MoveCursor(4,19); WriteString("I"); PrintString("DI"); PrintChar(CR);
        };
                if ((s==1) || (s==2) || (s==3))
```

```
                {
                MoveCursor(4,19); WriteString("JI"); PrintString("DJI"); PrintChar(CR);
                 };
                if (s==4)
                {
                MoveCursor(4,19); WriteString("J"); PrintString("DJ"); PrintChar(CR);
                 };
                if ((s==5) || (s==6) || (s==7))
                {
                MoveCursor(4,19); WriteString("JZ"); PrintString("DJZ"); PrintChar(CR);
                };
                if (s==8)
                {
                MoveCursor(4,19); WriteString("Z"); PrintString("DZ"); PrintChar(CR);
        };
                if ((s==9) || (s==10) || (s==11))
                {
                MoveCursor(4,19); WriteString("SZ"); PrintString("DSZ"); PrintChar(CR);
                };
                if (s==12)
                {
                MoveCursor(4,19); WriteString("S"); PrintString("DS"); PrintChar(CR);
           };
                if ((s==13) || (s==14) || (s==15))
                {
                MoveCursor(4,19); WriteString("SI"); PrintString("DSI"); PrintChar(CR);
           };
         };


}
//-------------------------------------------------------

void PrikazTlak (unsigned int t)
{

        if (t < 900) t = 900;
        if (t == 1167) t =0;
        if (t > 1050) t = 1050;


        PrintChar('P'); PrintInt(t, 4); PrintChar(CR);
        MoveCursor(2,7); WriteString("            "); MoveCursor(2,7);
        WriteInt(t, 4); WriteString("hPa");
}
//-------------------------------------------------------

void PrikazVlaga (unsigned int v)
{
        if (v < 0) v = 0;
```

```
        if (v > 100) v = 100;
        PrintChar('H'); PrintInt(v, 2); PrintChar(CR);
        MoveCursor(3,16); WriteString("     "); MoveCursor(3,16);
        WriteInt(v,2); WriteString("%");
}
//-------------------------------------------------------

void InitPrikaz(void)
{
        InitRS232();
        InitLCD();
        ClearLCD();
        HideCursor();
        MoveCursor(1,1); WriteString("Temperatura: ");
        MoveCursor(2,1); WriteString("Tlak: ");
        MoveCursor(3,1); WriteString("Rel. vlaznost: ");
        MoveCursor(4,1); WriteString("Vjetar: ");
}
//-------------------------------------------------------

unsigned int MjerenjeBrzina(void)
{

        Tint = 100; sec = 0; nsec = sec; BrImp = 0;
        SetTimer(2,CTimer);
        StartTimer(2);
  BrSig=1;
        while (sec < odbrojavanje)
        {
                if (BrSig == 0)
                {
                  BrImp++;
                  while (BrSig==0);
                };
        };
        StopTimer(2);
        BrImpF = BrImp * KoefBr;
        BrImpF = BrImpF *10;
        BrImpF = BrImpF + 0.5;
        BrImpI= (int)BrImpF;
        return (BrImpI);
}
//-------------------------------------------------------

unsigned int MjerenjeSmjer(void)
{
        data unsigned int stanje, S1, S2, S4, S8;

        P1 = 0xE1;
        if (stanje1 == 1) S1 = 1;
```

```
        if (stanje1 == 0) S1 = 0;
        if (stanje2 == 1) S2 = 2;
        if (stanje2 == 0) S2 = 0;
        if (stanje4 == 1) S4 = 4;
        if (stanje4 == 0) S4 = 0;
        if (stanje8 == 1) S8 = 8;
        if (stanje8 == 0) S8 = 0;
        P1 = 0xFF;
        stanje = S1+S2+S4+S8;
        return(stanje);
}

//-----------------------------------------------

unsigned int MjerenjeTlak(void)
{
        data unsigned int vrijednost, tlakI;
        data float tlakF;

        CLKP = 1; CSP = 1; Dout = 1;
        vrijednost = LTC1298(1);
        tlakF = (vrijednost-411.5)/3.184 + KalibTlak;
        tlakF = tlakF + 0.5;
        tlakI= (int)tlakF;
        CLKP = 0; CSP = 0; Dout = 0;
        return(tlakI);
}

//-----------------------------------------------
main()
{
        InitPrikaz();

        Mjerenje_Temp_i_Vlage();
        PrikazTemp(temp);
        Delay(1000);

        tlak = MjerenjeTlak();
        PrikazTlak(tlak);
        Delay(1000);

        PrikazVlaga(vlaga);
        Delay(1000);

        brzina = MjerenjeBrzina();
        PrikazBrzina(brzina);
        Delay(1000);

        smjer =  MjerenjeSmjer();
        PrikazSmjer(smjer);
```

```
        Delay(5000);

        while (1)
        {
                for (iter = 0; iter < 8; iter++)
                {
                        brzina = MjerenjeBrzina();
                        PrikazBrzina(brzina);
                        Delay(1000);

                        smjer =  MjerenjeSmjer();
                        PrikazSmjer(smjer);

                        Delay(5000);
                }
                Mjerenje_Temp_i_Vlage();
                PrikazTemp(temp);
                Delay(1000);

                tlak = MjerenjeTlak();
                PrikazTlak(tlak);
                Delay(1000);

                PrikazVlaga(vlaga);
                Delay(1000);
        };
}
```