VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

INFORMATIKOS KATEDRA

Paskirstyta mikroprocesorinių valdiklių sistema

Distributed microcontrollers system

Magistro baigiamasis darbas

Atliko:	Andrius Radzevičius	(parašas)
Darbo vadovas:	Dr. Mindaugas Viliūnas	(parašas)
Recenzentas:	Viktor Kiško	(parašas)

Vilnius – 2009

Santrauka

Magistro baigiamojo darbo tema – paskirstyta mikroprocesorinių valdiklių sistema.

Šio darbo tikslas buvo suprojektuoti, trikdžiams atsparią, paskirstytų mikroprocesorinių valdiklių sistemą.

Darbe apžvelgiamos mikroprocesorinių valdiklių duomenų perdavimo technologijos, mikroprocesoriniai valdikliai, mikroprocesorinių valdiklių sistemos.

Darbo metu buvo suprojektuota ir pagaminta trikdžiams atspari mikroprocesorinių valdiklių sistema.

Taip pat šiame darbe aprašomas paskirstytos mikroprocesorinių valdiklių sistemos mazgų programavimo procesas.

Atlikus baigiamąjį magistro darbą gautos tokios išvados: perteklinių funkcijų panaudojimas, paskirstytų mikroprocesorinių valdiklių sistemų mazguose, padėjo padidinti tokios sistemos atsparumą trikdžiams bei užtikrinti stabilų sistemos darbą; paskirstytos mikroprocesorinių valdiklių sistemos be vedančiųjų mazgų pranašesnės už sistemas su vedančiuoju mazgu, tačiau tokioje sistemoje yra daug sudėtingiau užprogramuoti sistemos mazgus.

Raktiniai žodžiai: mikroprocesorinis valdiklis, paskirstyta sistema, CAN sąsaja, protokolas, mazgas, programavimas, atsparumas trikdžiams, projektavimas.

Summary

Master‘s final thesis subject is distributed microcontrollers system.

The objective point for this master thesis was to project fault tolerant distributed microcontrollers system.

In master thesis overlooked microcontrollers data transfer technologies, microcontrollers and systems of microcontrollers.

During this work was projected and made fault tolerant distributed microcontrollers system and programmed the nodes of this system.

This master thesis conclusions are: redundant functions in the nodes of distributed microcontrollers system, helped to increase system stable and resistant; comparing the system without master nodes with system with master nodes, system without master nodes is more advantaged then system with them but this system programming process is more complicated.

Keywords: microcontroller, distributed system, CAN bus, protocol, node, programming proccess, fault tolerant, projecting.

Turinys

Įvadas

Šiais technologijų laikais retas žmogus įsivaizduoja savo gyvenimą be elektronikos prietaisų (televizoriaus, telefono, muzikinio centro, kompiuterio ir t.t.). Kiekviename iš minėtų prietaisų naudojami mikroprocesoriniai valdikliai.

Mikroprocesorinis valdiklis yra svarbiausia daugelio šiuolaikinių įrenginių dalis. Plačiausiai ir efektyviausiai mikroprocesoriniai valdikliai (MV) naudojami kontrolės, matavimo aparatūroje, tačiau vis daugiau jų diegiama valdymo sistemose, ryšių technikoje, medicinos diagnostikos ir gydymo aparatūroje, energetikoje, pramonėje bei roboto technikoje.

Naudojant mikroprocesorinius valdiklius matavimo aparatūroje, prietaisai tapo daug tikslesni, patikimesni. Taip pat išsiplėtė jų funkcinės galimybės, supaprastėjo matavimo proceso valdymas bei atsirado galimybė išvengti klaidų, kurios dažnai atsiranda dėl žmogaus kaltės.

Mikroprocesoriniai valdikliai yra specializuotos paskirties procesoriai. Pagal atliekamo uždavinio specifiką prie MV galima prijungti papildomus įrenginius pvz.: daviklius, indikatorius, jungiklius, vykdymo mazgus ir t.t. Tai praktiška ir aktualu, kai norima išplėsti MV galimybes, papildyti juos naujomis funkcijomis bei užtikrinti stabilų sistemos darbą.

Sparčiai tobulinant mikroprocesorinių valdiklių techniką MV tapo galingesni bei pigesni. Tačiau MV yra ribotų resursų įrenginiai, todėl norint realizuoti sudėtingus uždavinius viršijančius MV galimybes reikia projektuoti sistemas sudarytas iš daugelio tarpusavuje sujungtų MV. MV sujungimas realizuojamas naudojant aparatines ryšio sąsajas. Aparatinėmis ryšio sąsajomis atliekami informacijos mainai (tarp MV ir MV arba tarp MV ir kompiuterio) taip pat perduodami valdymo signalai. Duomenų mainai tarp mikroprocesorinių valdiklių vykdomi per lygiagrečiąsias ir nuosekliąsias sąsajas. Ryšio sąsajų yra daug ir įvairių, todėl projektuojant sistemą sudarytą iš daugelio MV svarbu susipažinti su populiariausiomis ir apžvelgti jų minusus bei pliusus. Daugeliui sąsajų egzistuoja bent po keletą aukštesnio lygio protokolų, todėl svarbu išanalizuoti protokolų panašumus skirtumus bei galimybes.

Sistemos sudarytos iš kelių mikroprocesorinių valdiklių gali būti centralizuotos arba paskirstytos, todėl svarbu apžvelgti tokių sistemų privalumus bei trūkumus. Šiuo metu pastebimas didelis susidomėjimas paskirstytomis sistemomis. Šiomis sistemomis domimasi todėl, kad sistema, sudaryta iš daugelio pigių komponetų, gali atlikti tokį pat valdymo ar informacijos apdorojimo uždavinį, kaip ir centralizuota, kurios kaina gali būti keliolika kartų aukštesnė. Tačiau paskirstytų sistemų valdymas yra daug sudėtingesnis nei centralizuotų. Todėl reikia panagrinėti paskirstytų sistemų valdymo metodus.

Mikroprocesorinių valdiklių sistemos dažnai naudojamos situacijose kuriose svarbu ne tik įvykdyti pavestą užduotį, bet ir svarbu ją įvykdyti per tam tikrą laiką. Tokios sistemos vadinamos realaus laiko sistemomis, tad reikia apžvelgti ir tokias sistemas bei jose naudojamas operacines sistemas mikroprocesoriniams valdikliams.

Apžvelgus aukščiau išvardintas technologijas bei pasirinkus tinkamas, galima suprojektuoti paskirstytą mikroprocesorinių valdiklių sistemą, kuri atliks tam tikrą darbą. Paskirstytų mikroprocesorinių valdiklių sistemų yra sukurta nemažai, tačiau šiame darbe projektuojama sistema išsiskiria tuo, kad sistemos darbas vyksta be valdančiojo įrenginio bei ji atspari trikdžiams. Trikdžių šaltiniais sistemoje gali būti:

- ryšio linijoje atsirandantys trikdžiai;
- ryšio linijos fiziniai pažeidimai;
- sistemą sudarančių mazgų gedimai, nesutrikdantys ryšio linijos darbo;
- mazgų komponentų neteisingas funkcionavimas, nesutrikdantis mazgo darbo;

Projektuojamoje sistemoje bus siekiama, kad sistema būtų atspari ryšio linijos trikdžiams, mazgų atsijungimams, prisijungimams, mazgų sudedamųjų dalių gedimams. Sistemos atsparumas pasiekiamas, sutrikusio mazgo funkcionalumą perdavus mazgams turintiems perteklinį funkcionalumą. Todėl tokio tipo sistema yra stabilesnė bei atsparesnė gedimams lyginant su tomis sistemomis, kur yra valdantysis mazgas.

Tokiose aplinkose, kuriose būtina užtikrinti nepertraukiamą sistemos darbą, tokios sistemos yra reikalingos ir aktualios.

Šio darbo tikslas – suprojektuoti, trikdžiams atsparią, paskirstytų mikroprocesorinių valdiklių sistemą.

Darbo uždaviniai:

1. ***Technologijų analizė:***

- apžvelgti ryšio sąsajas bei palyginti jų galimybes;
- apžvelgti geriausiai tinkančios ryšio sąsajos mikroprocesorinių valdiklių sujungimui protokolus.
- apžvelgti mikroprocesorinius valdiklius;
- apžvelgti paskirstytas mikroprocesorinių valdiklių sistemas;
- apžvelgti realaus laiko sistemas;
- apžvelgti realaus laiko operacines sistemas;

2. ***Sistemos projektavimas:***

- pagal technologijų analizę pasirinkti tinkamiausias technologijas;
- pagal pasirinktus sistemos komponentus suprojektuoti sistemos mazgų principinę schemą.

- suprojektuotai sistemai reikia atlikti trasavimą ir padaryti spausdintines plokštes;
- sujungti sistemos komponentus (komponentų sumontavimas);
- atlikti testavimą.

3. Sistemos programavimas:

- pasiriktai sąsajai parinkti aukštesnio lygio protokolą;
- papildyti protokolą taip, kad tenkintų sistemos reikalavimus
- sudaryti mazgų darbo algoritmą;
- užprogramuoti sistemos mazgus;
- sistemos sekimui suprogramuoti stebėjimo sistemą.

4. Ištirti sistemos elgseną tuomet, kai yra sutrikdomas sistemos darbas.

- Atlikti eksperimentus ir pakoreguoti sistemos algoritmą taip, kad sistemos darbas nesutriktų atsijungus vienam ar keliems mazgams bei atsijungusių mazgų funkcionalumą perimtų mazgai turintys perteklinį funkcijų kiekį

Laukiami darbo rezultatai yra tokie:

- suprojektuota matavimo sistema bei tinkamai parinktos sistemos sudedamosios dalys;
- realizuota matavimo sistema su MV ir nuosekliąja sąsaja, įvykdant visus reikiamus technologinius procesus bei parenkant tinkamas priemones;
- užtikrintas sistemos mazgų algoritmų darbas esant ryšio ir komponentų sutrikimams;
- gedimų fiksavimo ir perdavimo į kopiuterį mechanizmas.

1. Duomenų perdavimo technologijų apžvalga

1.1 Aparatinės ryšio sąsajos

Norint išsirinkti vieną iš sąsajų iš pradžių reikia išsiaiškinti:

- kas yra aparatinės ryšio sąsajos ir kokie jų tipai;
- aparatinę magistralinio ryšio sąsajų klasę „Fieldbus“;
- palyginti „Fieldbus“ sąsajų klasės tipus ir pasirinkti tinkamiausią.

Aparatinė ryšio sąsaja - tai ryšio ir sąveikos priemonių tarp mikroprocesorinių komponentų visuma. Sąsajos gali būti daugiakontūrinės arba magistralinės.

Paprasčiausias būdas keistis duomenimis – du komunikacijos įrenginius jungti laidų linija. Tokiu būdu jungiant daug įrenginių, gaunamas daugiakontūris tinklas: kai yra n tinklo mazgų, kiekvienas mazgas turi turėti (n-1) jungimo linijų. [Dek00] Toks tinklo jungimas neoptimalus bei daug kainuoja. Kitas jungimo būdas gali būti magistralinis.

Magistralinėje struktūroje informacijos mainams visi tinklo mazgai naudoja bendrą perdavimo liniją. Prie bendros magistralės tinklo mazgai prijungiami atšakų linijomis. Šių sąsajų privalumai: reikia mažiau kabelinių linijų ir kiekvienam tinklo mazgui būtinas tik vienas prijungimo portas. Trūkumas- vienu metu duomenis gali siųsti tik vienas tinklo mazgas. Todėl jai turi būti nustatytos tinklo užvaldos taisyklės, nes tik užvaldęs magistralę tinklo mazgas įgauna teisę siųsti duomenis. Be to, vienam prietaisui užvaldžius magistralę ir sutrikus prietaiso darbui, sutrikdomas visas sistemos darbas, tačiau norint išvengti tokios situacijos kiekvieno įrenginio informacijos perdavimui skiriamas tam tikras laiko tarpas.

Aparatinė magistralinio ryšio sąsajų klasė (Fieldbus)- tai dvipusė, daugelio pajungimo taškų, pramoninė ryšio sąsajų klasė, kuri naudojama, atskirų ir apdorojimo, įrenginių sujungimui valdymo sistemose [Gof00]. FieldBus tuo pačiu laidu gali perduoti diagnostinius, matavimo duomenis ir valdymo komandas. FieldBus tinklo struktūra gali būti įvairi : žvaigždės, žiedo, šakos ar medžio formos. Kadangi prie FieldBus sąsajos vienu metu gali būti prijungta daug prietaisų, todėl ji yra daugiafunkcinė [Gof00]. FieldBus- tai pirma visiškai paskirstyta sąsajų klasė.

Šiuo metu galima rinktis iš daug FieldBus tipo sąsajų, kurias sistemos projektuotojas gali pasirinkti pagal tam tikrus kriterijus. Toliau nagrinėsiu tokias Fieldbus tipo sąsajas: CAN, LIN, I2C, MODBUS, PROFIBUS-DP. Norint tinkamai pasirinkti FieldBus sąsają, reikia atkreipti dėmesį į tam tikrus reikalavimus: integraciją su egzistuojančia įranga, pageidaujamą funkcionalumą.

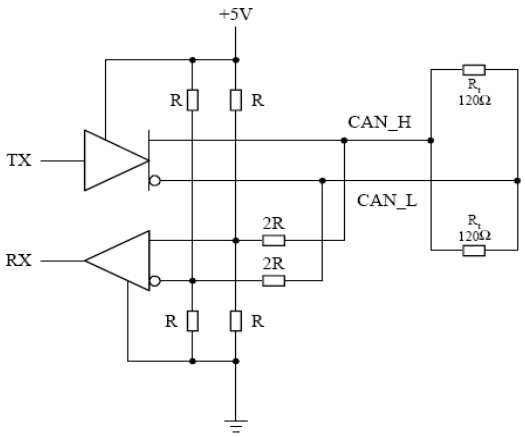
Norint užtikrinti optimalų ryšio linijų darbą, sąsajos vertinamos pagal tokius kriterijus:

- duomenų pralaidumą (greitį);
- didžiausią atstumą;
- patikimumą (aparatinės realizacijos lemtas; aparatinis klaidų aptikimas ir taisymas);
- maksimalų vienoje magistralėje dirbančių įrenginių kiekį;
- ar yra vedantis įrenginys, ar jo funkcija yra pereinama.
- kainą;
- protokolų realizacijos paprastumą.

Nuosekliosios sąsajos daugeliu parametrų pranašesnės už lygiagrečiąsias t.y. mažesnis jungiamų laidų kiekis, dėl mažesnio laikiškai suderintų linijų kiekio galimas didesnis perdavimo atstumas pasiekiama didesnė perduodamos informacijos sparta, taip pat prie nuosekliųjų sąsajų galima prijungti didelį kiekį įrenginių bei nuosekliosioms sąsajoms realizuoti reikalingos mažesnės piniginės sąnaudos.

1.1.1 CAN sąsaja

CAN (Controller Area Network) - tinklo sąsaja, skirta mikroprocesorinių valdiklių ir sensorių sujungimui į vieną tinklą. CAN sąsają sukūrė kompanijos Intel Corporation ir Robert Bosch GmbH. Pradžioje ji buvo skirta automobilių pramonei, kuriai reikėjo patikimų ir trikdžiams atsparių komunikacijų [Tiar02]. Šiuo metu CAN sąsaja naudojama daugelyje sričių: pramonės automatizacijoje, robotuose bei stebėjimo sistemose. Sąsajos principinė schema pateikta 1 pav.



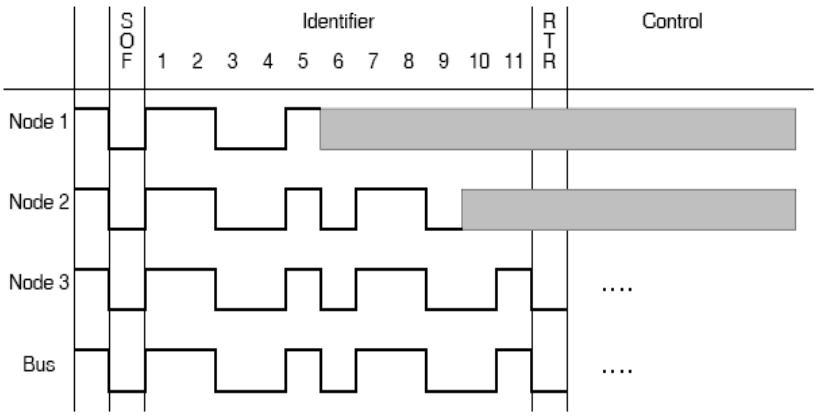
1 pav. CAN sąsajos principinė schema. [Tiar02]

Pagrindinės CAN sąsajos savybės:

- ji palaiko iki 1 Mbits/s<=40m greitį;

- maksimalus atstumas tarp mazgų gali būti iki 1km;
- perdavimo tipas- diferencialinis,o ryšio linijos tipas: vyta pora (STP), (UTP);
- kadangi tai diferencialinis linijos tipas, tai loginis „1“ = 0 V(skirtumas tarp CAN_H ir CAN_L kiekvieno atskirai 2.5 V) ir loginis „0“ = 2 V (skirtumas tarp CAN_H = 3.5 ir CAN_L = 1.5 V) . Iėjime fiksuojamas didesnis kaip 0.2 V skirtumas. Loginių būsenų (loginio 0, 1) įėjimo histerezė 100 – 180 mV.
- iki 256 mazgų vienoje linijoje;
- objekto vardui skiriama arba 11 bitų (CAN 2.0A, ISO 11898:1993) arba 29 bitai (CAN 2.0B, ISO 11898:1995)
- maksimalus duomenų kiekis viename pranešime 8 baitai (ilgio ribojimas neleidžia vienam mazgui ilgai užimti magistralę).
- aparatinė klaidų kontrolė;
- ryšio kanalinis sluoksnis (data link layer) OSI (abstraktus ryšio protokolų, naudojamų ryšio ir kompiuteriniuose tinkluose aprašymas) modelyje yra standartizuotas ISO 11898-1 (2003m.)
- fizinis sluoksnis standartizuotas: ISO 11898-2 (didelio greičio), ISO 11898-3 (mažo greičio atsparus gedimams), o kitų sluoksnių realizacija yra palikta laisvam sistemos projektuotojo pasirinkimui.

Duomenų mainai. Sąsajos veikimas pagrįstas pranešimų siuntimu iš įrenginio i magistralę. Kiti įrenginiai savo ruožtu atsiliepia ne į visus pranešimus, bet tik į tuos, kurie skirti jiems. Kai sąsaja laisva (nevyksta duomenų mainai tarp valdiklių), bet kuris iš prijungtų valdiklių gali pradėti duomenų siuntimą. Situacija, kai keli mazgai nori siųsti duomenis vienu metu, vadinama arbitražu. Šią situaciją iliustruoja 2 pav. Iš paveikslėlio matyti, kad kiekvieno mazgo identifikatorius yra toks: Node1 = 11001101010, Node2 = 11001011011, Node3 = 11001011001. Pradžioje kiekvienas iš mazgų siunčia kadro pradžios bitą (SOF), o po to siunčiami identifikatoriaus bitai tol, kol visų trijų mazgų bitai sutampa. Kai susidaro tokia situacija, kai bitai nesutampa, lieka siųsti tik tie įrenginiai, kurių siunčiamas bitas yra dominuojantis (CAN sąsajoje „0“ pasirinkta dominuojanti būsena). Objekto vardas kartu nustato ir jo prioritetą, kur mažesnės identifikatoriaus reikšmės turi prioritetą prieš didesnes. Kaip matome iš 2 pav. siųsti lieka trečias mazgas, o pirmas (iškrenta po 6 bito) ir antras (iškrenta po 10 bito) įrenginiai pereina į priėmimo būseną.

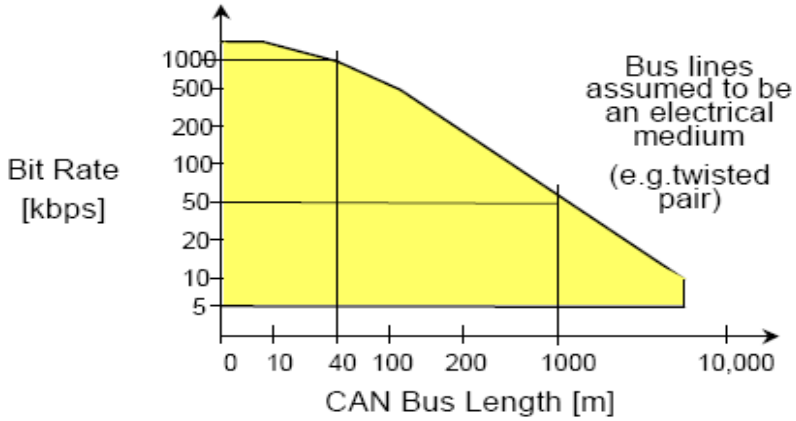


2 pav. Arbitražo procedūra CAN sąsajoje su 3 mazgais [JTN05].

CAN sąsajoje realizuota dviejų pakopų klaidų kontrolė. Kiekvieną pranešimą turi patvirtinti bent vienas priimantis mazgas. Tam tikslui, pranešimo formate numatytas kvitavimo bitas, kurį (pranešimą teisingai priėmę mazgai) nustato į “0”, t.y. į dominavimo būseną. Kadangi tai atliekama dar pranešimo perdavimo metu, todėl labai greitai atpažįstama duomenų siuntimo klaida. Atpažinus klaidą, siųstuvas kartoja pranešimą tol, kol gaunamas pakeistas kvitavimo bitas. Praktiškai konkrečiuose protokolo taikymuose daroma taip: kad klaidingas pranešimas neblokuotų visos magistralės, ribojamas pranešimo kartojimų skaičius. Konkrečiais atvejais, naudojami mechanizmai, kuriuose sugedę mazgai patys atsijungia nuo tinklo.

Sąsajos privalumai: darbas realiame laike, paprasta realizacija, minimalios piniginės sąnaudos, geras klaidų valdymas siuntimo ir priėmimo rėžimuose, platus greičių darbo diapazonas, didelis paplitimas. Sąsajos trūkumas- didelis kiekis tarnybinės informacijos pakete.

Taip pat svarbi sąsajos savybė, kad linijos ilgis atvirkščiai proporcingas greičiui (CAN sąsajos greičio priklausomybė nuo sąsajos ilgio pateikta 3 pav.). Į šią savybę svarbu atsižvelgti projektuojant sistemą bei parenkant atstumus tarp sistemos mazgų.



3 pav. CAN sąsajos greičio priklausomybė nuo sąsajos ilgio [SIE96].

CAN sąsaja buvo projektuojama atsižvelgiant į OSI modelį, kuris naudojamas kompiuteriniuose tinkluose. Ši sąsaja yra pakankamai pajėgi naudoti tokius standartinius kompiuterinių tinklų protokolus kaip TCP/IP. Galbūt kažkada ateis tokia diena, kai internetu galėsime įjungti mašinos šildymą ankščiau negu paliksime savo namus [SB02].

1.1.2 LIN sąsaja

LIN (Local interconnect network) sąsaja suprojektuota LIN konsorciumo, o pirma specifikacija buvo publikuota 1999 metais [Rey03]. LIN tinklo sąsaja pasižymi nedideliu greičiu ir yra naudojama kaip pigi CAN sąsajos posistemė, kuri sujungia daviklius bei pavaras. Šiuo metu naudojama 2.1 sąsajos versija. LIN sąsają galima sujungti, panaudojant elektros perdavimo liniją, bet tam būtinai reikalingas specialus DC-LIN siųstuvas- imtuvas.

Pagrindinės LIN sąsajos savybės:

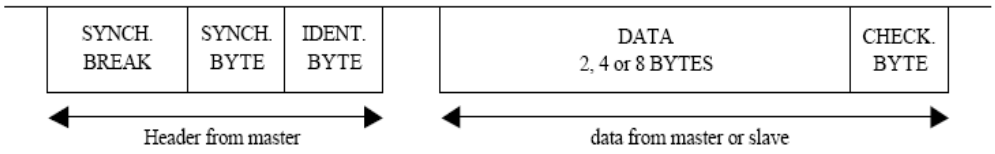
- greitis 1-20kbit/s;
- maksimalus atstumas tarp mazgų 40m;
- ryšio linijos tipas: 1 laido magistralė;
- signalų charakteristikos: loginis „1”>=80%VDD; loginis “0” <=20%VDD; VDD=8..18V; (VDD-maitinimo įtampa)
- iki 15 valdomųjų įrenginių;
- skiriami du įrenginiai - vedantysis (master) ir vedamasis (slave);
- multicast (broadcast) pranešimai;
- nuosava sinchronizacija (tik valdantysis įrenginys turi savą laikrodį);
- pranešimai su 2,3 ar 8 duomenų baitais, 3 kontrolės baitai;
- klaidų aptikimas (8bitų kontrolinė suma ir 2 pariteto bitai identifikatoriuje)
- fizinis sluoksnis standartizuotas ISO9141;
- budėjimo rėžimo palaikymas.

Specifikacija apibūdina 3 iš 7 OSI modelio sluoksnių fizinį , ryšio kanalinį ir taikomąjį sluoksnį.

LIN yra nuoseklioji sąsaja, sudaryta iš vieno valdančiojo (master) ir iki 15 valdomųjų (slave) įrenginių. Sąsajos jungimo schema pateikta 4 pav. Visus pranešimus inicijuoja valdantysis įrenginys ir jis keičiasi pranešimais tik su vienu valdomuoju įrenginių vienu metu. Paprastai valdantysis mikroprocesorinis valdiklis yra daug galingesnis už valdomuosius mikroprocesorinius valdiklius.

4 pav. LIN sąsajos jungimo schema [Rey03].

Protokolas. LIN valdantysis įrenginys valdo visas duomenų užklausas, perdavimus bei siunčia pranešimo antraštes valdomiesiems įrenginiams. Valdomieji įrenginiai negali siųsti antraštės (header), o gali tik gražinti pranešimo atsakymą. Valdantysis įrenginys taip pat gali formuoti atsakymą. Įrenginiai keičiasi pranešimais, kurie sudaryti iš antraštės (header) ir atsakymo (response).



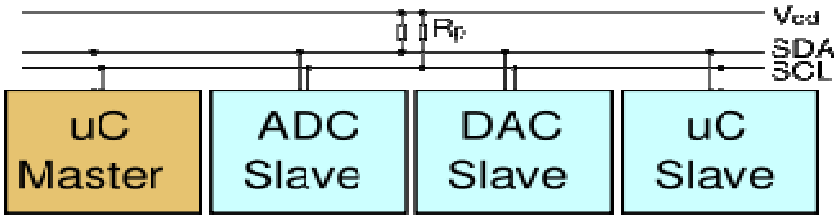
5 pav. LIN sąsajos pranešimo struktūra [Rey03].

Pranešimo struktūra: Synchronization break, sinchronizacijos baitas (paskirtis valdomojo įrenginio sinchronizacijai), identifikatoriaus baitas, duomenų baitai, kontrolinės sumos baitas. Pranešimo struktūra pateikta 5 pav.

Klaidų aptikimas. LIN sąsaja daugumos klaidų neaptinka. Tokios klaidos kaip: kontrolinės sumos klaida, identifikatoriaus pariteto klaida, valdomojo įrenginio neatsakymas, blogas sinchronizacijos kadras ir sąsajos miego būsena, gali būti aptiktos, tačiau tokiu atveju jas turi valdyti valdantysis įrenginys.

1.1.3 I2C sąsaja

I2C (inter integrated circuit) sąsaja kartais dar vadinama TWI –two Wire interface (kad nereiktų mokėti už licenziją). Tai dvilaidė (trečias laidas- žemė) sinchroninė sąsaja (CAN, LIN asinchroninės sąsajos), sukurta Philips firmoje [Phi07]. Įrenginiai čia sujungiami dviem signalais: SCLOCK ir SDATA. Sąsajos jungimo schema pateikta 6 pav. Ši sąsaja sudaryta iš tokių įrenginių: valdančiojo (master) ir valdomojo (slave). Signalas SCLOCK naudojamas perdavimo ir priėmimo sinchronizacijai, o signalas SDATA – dvikrypčiams duomenų mainams. Signalai SCLOCK ir SDATA yra dvikrypčiai.



6 pav. I2C sąsajos jungimo schema. [Phi07]

Pagrindinės I2C sąsajos savybės:

- greitis 10 kbit/s – 400 kbit/s; (standartai 100 kbit/s, 400 kbit/s)
- maksimalus atstumas tarp mazgų 10m;
- ryšio linijos tipas: vyta pora; 4-laidis;
- signalų charakteristikos: loginis „1”<=0.3VDD; loginis “0” >=0.7VDD; VDD=5V (dažniausiai) (VDD-maitinimo įtampa);
- iki 112 vedamųjų įrenginių (adresų erdvė 7 bitai – 16 rezervuotų);
- skiriami du įrenginiai - vedantysis (master) ir vedamasis (slave). Vedantysis įrenginys gali būti ir kintantis.

Prie sąsajos linijų gali būti prijungta keletas įrenginių. Mainai prasideda nuo “Start” bito, kurį generuoja vedantysis įrenginys. “Start” būsena prasideda pagal krintantį signalo SDATA frontą, kai signalas SCLOCK yra loginio vieneto būsenos. Po “Start” būsenos vedantysis įrenginys siunčia per liniją SDATA vedamajam įrenginiui baitą (vyriausiuoju bitu į priekį). Čia yra vedamojo įrenginio adresas ir skaitymo ar rašymo (R/W) būsenos bitas. Pirmi septyni bitai nusako vedamojo įrenginio adresą. Aštuntasis bitas parodo pranešimo perdavimo kryptį: “0” rodo, kad vedamasis įrenginys perduoda vedančiajam, “1” – kad priims iš vedančiojo. Priėmę adresą, visi vedamieji įrenginiai palygina savo adresą su priimtuoju. Jei šie sutampa, vedantysis įrenginys perduoda patvirtinimą. Jei patvirtinimas nėra gautas, vedantysis įrenginys generuoja būseną “STOP”.

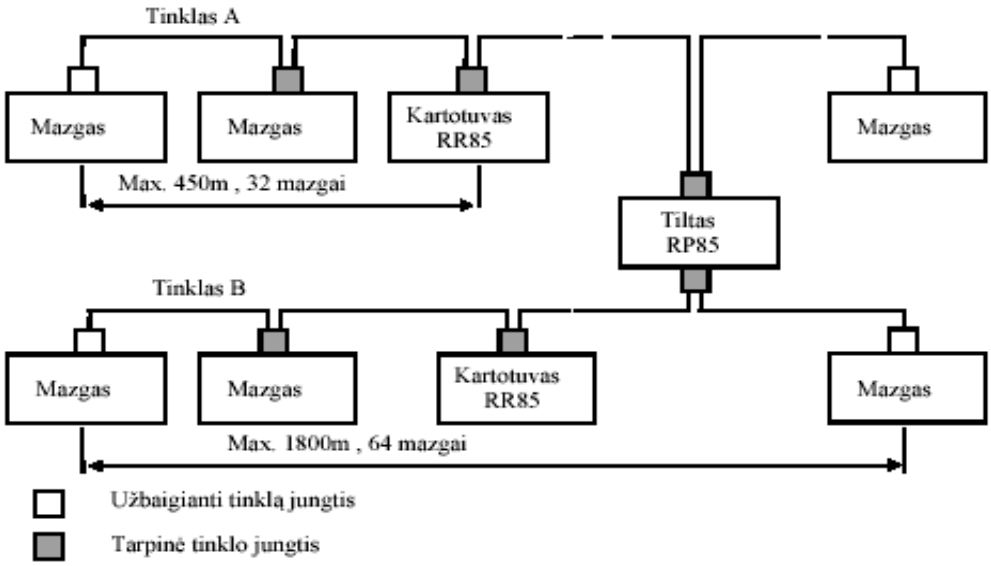
1.1.4 MODBUS sąsaja

MODBUS – tai nuoseklioji sąsaja. Duomenys perduodami susuktos laidų poros ryšio linija. MODBUS gali būti naudojamas: valdiklių duomenų apsikeitimui, valdiklių programavimui, duomenų perdavimui tarp valdiklių ir valdančiųjų kompiuterių, vartotojo programų įrašymui į valdiklius arba nuskaitymui iš jų. MODBUS sąsajos jungimo schema gali būti tokia kaip pavaizduota 7 pav.

Pagrindinės MODBUS sąsajos savybės:

- greitis iki 1 Mbit/s;
- maksimalus atstumas tarp mazgų- 450m;
- ryšio linijos tipas: vyta pora;
- signalų charakteristikos: loginis „1”= +3V; loginis “0” =-3V;
- tinkle gali būti iki 64 mazgu;
- du adresavimo būdai;

Modbus tinkle naudojama informacinio žiedo struktūra. Valdymo funkcijas perduodantis pranešimas- požymis žiede- vadinamas „token“. Kai tinklo mazgas gauna požymį, jis gali inicijuoti duomenų skaitymą/rašymą ir ryšį su kitais tinklo mazgais. Kai mazgas siunčia požymį, jis gali įrašyti į visiems mazgams priklausančią globalią duomenų bazę. Dėl to, bet kurio mazgo būsenos pasikeitimas tampa greitai žinomas kitiems mazgams.



7 pav. MODBUS sąsajos jungimo schema. [Dek00]

Keletą tinklų, įrenginių galima sujungti tarpusavyje. Šiuo atveju adresuojama daugiau kaip 64 mazgai, todėl Modbus tinkle numatyti du adresavimo būdai: tiesioginis ir netiesioginis. Kiekvieno pranešimo Modbus tinkle adresavimui skiriami 5, vienas paskui kitą einantys, baitai. Kai komunikacijos vyksta per MODICON valdiklio Modbus portą, tai vieno baito adresas perskaičiuojamas į Modbus 5 baitų adresą ir įjungiamas į pranešimą [MM96]. Adreso struktūra Modbus tinkle priklauso nuo paskirties mazgo tipo: jei Modbus pranešimo adresas yra intervale nuo 1 iki 64, tai pranešimas atitinkamam Modbus mazgui perduodamas be pakitimų. Toks adresavimas vadinamas betarpišku. Taip adresuojant, perdavimo kadre yra tik vienas nenulinis adreso baitas, tačiau taip adresuojant negalima pranešimo perduoti už lokalaus tinklo ribų. Jei Modbus pranešimo adresas yra intervale 70...79, tai valdiklis atlieka tiesioginį prijungimą, palygindamas Modbus adresą su lentele, saugoma valdiklyje.Taip adresuojant , nenuliniai gali

būti visi 5 baitai arba tik dalis jų, todėl Modbus pranešimus galima siųsti ne daugiau kaip per 4 BP85 tiltus, kurie jungia 5 Modbus tinklus. Jeigu Modbus pranešimo adresas yra intervale 80...255, tai valdiklis atlieka netiesioginį adresavimą. Šiuo atveju adresas dalijamas iš 10, o rezultatas ir liekana panaudojami kaip pirmas ir antras Modbus adreso baitai. Netiesiogiai adresuojant, galima pasiųsti pranešimą per vieną BP85 tiltą, kuris jungia 2 lokalius Modbus tinklus. Duomenų mainų organizavimui pasirenkamas reikalavimo pranešimo ir atsakymo pranešimo principas. Apsikeitimas pranešimais vyksta pusiau duplekso būdu: valdantysis prietaisas gali siųsti naują reikalavimo pranešimą, jeigu gavo atsakymą į prieš tai pasiųstą pranešimą, arba praėjo tam tikras nustatytas laikas, bet atsakymas iš pavaldaus prietaiso negautas. Pranešimo tipas nulemia, ar siunčiamame pranešime yra duomenų. Kai rašymo pranešimas yra su paklausimo pranešimu, tuomet perduodami duomenys, o atsakymo pranešimu tik patvirtinama, kad duomenų perdavimas atliktas. Tačiau, esant skaitymo pranešimui, su atsakymo pranešimu valdančiajam prietaisui persiunčiami pareikalauti duomenys. Duomenų mainų būdas nustatomas funkcijos kodu, kuris yra paklausimo pranešime. Pavaldusis prietaisas į iškraipytą arba klaidingą paklausimo pranešimą neatsako. Tada valdančiojo prietaiso uždavinys, praėjus nustatytam laikui, reikalavimą pakartoti. Jeigu pavaldusis prietaisas gavo teisingą paklausimo pranešimą, tačiau reikiamų duomenų visai negali arba tik šiuo metu negali pasiųsti, tada atsakoma specialiu pranešimu (Exception Response), kuriuo informuojama apie duomenų neatsiuntimo priežastį. Šiuo atveju, valdančiojo prietaiso uždavinys- spręsti apie tolimesnius duomenų mainus. Jeigu procesorius turi keletą Modbus portų, tai jie dirba vienas nuo kito nepriklausomai. Į portus ateinantys reikalavimo pranešimai praktiškai aptarnaujami tuo pačiu metu.

1.1.5 PROFIBUS-DP sąsaja

PROFIBUS – DP sąsajos paskirtis- žemesniųjų valdymo sistemos lygių sujungimas į tinklą ir informacijos perdavimas gamybos administravimo lygio tinklinei struktūrai. [ED02]

Pagrindinės savybės:

- greitis iki 12 Mbit/s;
- maksimalus atstumas tarp mazgų 1200m;
- ryšio linijos tipas: vyta pora;
- signalų charakteristikos: loginis „1<=25%VDD; loginis “0” >=75%VDD; VDD= 7..12V(VDD-maitinimo įtampa);
- tinkle gali būti iki 32 mazgu;
- 2 tipų valdantieji įrenginiai: PROFIBUS -DP ir PROFIBUS-FMS;

Pagrindinis PROFIBUS -DP išskirtinumas yra tas, kad jame proceso duomenys ciklišk ai pateikiami vartotojui. Vietoj modelio septintojo lygio, ryšiui su antruoju modelio lygiu sudaryti naudojamas standartinis vartotojo interfeisas bei tiesioginis duomenų komponavimas DDLM (Direct data link mapper). PROFIBUS -DP išskiriami du valdančiųjų įrenginių tipai - pirmos klasės valdantieji įrenginiai, komunikacijų procesui valdyti, ir antros klasės valdantieji įrenginiai- komunikacijų instaliavimui ir diagnozei atlikti. Grynai PROFIBUS– DP sistemose instaliuojami tik pirmos klasės valdantieji įrenginiai, tačiau mišriose sistemose gali būti naudojami PROFIBUS-DP ir PROFIBUS-FMS valdantieji įrenginiai.

1.1.6 ETHERNET sąsaja

Ethernet sąsają 1975 m. sukūrė *Xerox* (*Xerox,* Palo Alto Research Center – PARC) mokslinė grupė, vadovaujama Bob Metcalf.

Pagrindinės savybės:

- greitis iki 1Gb/s;
- maksimalus atstumas tarp mazgų 100 m (priklausomai nuo standarto ir kabelio tipo 10BaseFL su optiniu kabeliu);
- ryšio linijos tipas: Koaksialinis kabelis, 4-laidų vyta pora;
- signalų charakteristikos: loginis „1<=10%VDD; loginis “0>=90%VDD; VDD=2.5...3.3V;
- mazgų kiekis virtualiai neribojamas (vyta pora duoda tik 1:1 galimybę);

Ethernet tinklas gali būti konstruojamas naudojant žvaigždės ir magistralės topologijas (jei naudojamas vytos poros kabelis – Ethernet’as konfigūruojamas tik kaip žvaigždė). Šiuolaikinė Ethernet’o versija priimta 1982 m. Standartinis Ethernet’o produktyvumas – 10 Mbit/s arba 100 Mbit/s. Ethernet pagrindas yra užimtumo aptikimo arbitražo metodas. Ethernet informacinio paketo struktūra pateikta 1 lentelėje.

1 lentelė. Ethernet paketo struktūra. Laukų ilgiai pateikti baitais.

Preambulė	Paskirtis	Šaltinis	Tipas	Duomenys	CRC suma
8	6	6	2	46 – 100	4

Paketą sudaro šie elementai:

- ***preambulė***, skirta paketo sinchronizacijai, kuri sudaryta iš nulių ir vienetų sekų: pirmi 7 baitai yra vienodi – 10101010 ir tik 8-as baitas 10101011 – nesimetrinis;
- ***paskirtis ir šaltinis*** – tai unikalūs adresato ir siuntėjo 48 bitų adresai, paprastai

užrašomi šešioliktainiais skaičiais, pvz., 00 60 48 EC B2 C6. Šie adresai paskirstomi tinklo adapterių gamintojams ir fiksuojami adapterio pastoviojoje atmintyje, t. y. kiekvienas adapteris nuo pagaminimo momento jau turi savo identifikatorių – 48 bitų adresą;

- ***tipas*** (2 baitai) buvo įvestas *Xerox* ir naudotas vidinėms firmos reikmėms– aukštesnio lygmens protokolui nurodyti. Ethernete tipas neinterpretuojamas. Aukštesnio lygmens protokolai pagal tipą gali atpažinti paketą, nenagrinėdami paketo turinio, tai yra nelįsdami į ne "savo" paketus;
- ***duomenys***- šis laukas negali būti trumpesnis, nei 46 baitai;
- ***CRC*** – perteklinės ciklinės sumos liekana (Cyclic Redundancy Checksum) – kontrolinė suma, skaičiuojama, naudojant CRC - 32 ar kitokio tipo polinomus. CRC naudojama perduodamos informacijos klaidų kontrolei.

Ethernet technologiją apibendrina 1985 m. paskelbtas IEEE 802.3 standartas, bet reikia pabrėžti, kad IEEE 802.3 ir Ethernet šiek tiek skiriasi. Anksčiau sukurtas Ethernet IEEE 802.3 standartą tenkina nepilnai: pastarajame papildomai išskirti MAC ir LLC lygmenys, kurių nėra originaliojoje Ethernet versijoje bei atsisakyta Ethernet konfigūracijos testavimo protokolo ECTP (Ethernet Configuration Test Protocol). Vienas iš pagrindinių skirtumų yra besiskiriantys informacinių laukų tipai ir dydžiai.
IEEE 802.3 standartą atitinkančio informacinio paketo struktūra pateikta 2 lentelėje.

2 lentelė. IEEE 802.3 standarto paketo struktūra. Laukų ilgiai pateikti baitais.

Preambulė	Pradžios žymė	Adresatas	Siuntėjas	Ilgis	Duomenys	Balastas	CRC suma
7	1	6 arba 2	6 arba 2	2	0 –1500	Jei duomenų mažiau nei 46 baitai pridedamas balastas iki 46 baitų	4

Preambulę sudaro 7 baitai pasikartojančių nulių ir vienetų. Paketo pradžios žymė – 10101011. Adresato ir siuntėjo adresai gali būti pateikti 6 arba 2 baitais – sutrumpintas adreso variantas. Paskirties adrese gali būti išskiriamas pirmas bitas I/G (individualus ar grupinis adresas), kuris lygus 0, jei paketas skirtas konkrečiam gavėjui ir 1, jei paketas skirtas grupei. Ilgis nurodo duomenų baitų skaičių. Jei duomenų mažiau negu 46 baitai, pridedamas balastas (iki 46 B). Tiek Ethernet, tiek IEEE 802.3 paketo ilgis yra tarp 64 ir 1518 baitų (preambulė ir paketo pradžios žymė neįskaitomi). Populiariausias iš IEEE 802.x standartų – IEEE 802.3 dar

vadinamas 10BASE.

[Art02] Šiame straipsnyje buvo pademonstruotas paskirstytos namų valdymo sistemos pavyzdys, naudojantis ETHERNET sąsaja bei naudojant PIC mikroprocesorinius valdiklius. Taip pat parodyta, kad namų prietaisai, tokie kaip: skalbimo mašina, šviesos jungikliai, temperatūros davikliai, ventiliatoriai, gali būti kontroliuojami paskirstytos visuotinai pasiekiamos aparatinės įrangos. Tačiau perskaičius straipsnį galima daryti išvadą, jog prie kiekvieno įrenginio papildomai reikalinga tinklo plokštė ir kai tokių įrenginių kiekis didelis, tai gali sąlygoti didelę sistemos kainą ir sunaudojamą energiją.

1.1.7 Sąsajų palyginimas

Praeituose skyriuose apžvelgiau sąsajas, šiame skyriuje apibendrinsiu ir palyginsiu apžvelgtas sąsajas. Pasirikti kriterijai:

- greitis;
- atstumas tarp mazgų – kokiu nuotoliu gali būti išdėstyti sistemos mazgai, tai svarbu kai sistema būna išdėstyta nutolusiose vietose;
- ryšio linija – kokia naudojama ryšio linija t.y. koks kabelis;
- signalų charakteristikos ;
- galvaninis atskyrimas – leidžia eliminuoti vieno mazgo įtaką kitam per ryšio elektrinę grandinę;
- palaikomas mazgų skaičius;
- ar yra klaidų aptikimas;
- ar yra aparatinis klaidos atstatymas;
- ar egzistuoja integruotas protokolas – ar protokolas realizuotas aparatiškai t.y. yra integruotas į MV. Jei yra, kokia to įtaka į MV kainą;
- koks vidutinis naudingų duomenų santykis su visu persiųstų duomenų kiekiu ;
- ryšio linijos rėžimas - duomenų perdavimo kryptis;
- sinchronizacija.

Pagal pasirinktus kriterijus galima sudaryti sąsajų palyginimo lentelę. Tai 3 lentelė, kuri pateikiama žemiau.

3 lentelė. Sąsajų palyginimas

Požymis / **Sąsaja**		**CAN**	**LIN**	**I2C**	**MODBUS**	**PROFIBUS-DP**	**ETHERNET**
greitis	min	10kbit/s	1kbit/s	10kbit/s		1,5Mbit/s	10Mbit/s, 100 Mbit/s,
	max	1 Mbits/s<=40m	20kbit/s	400kbit/s	1 Mbit/s	12Mbit/s	1Gbit/s
atstumas tarp mazgų max		1000m	40m	10m	450m	1200m	100m
ryšio linija		Vyta pora (STP),(UTP)	Vienalaidė	Vyta pora	Vyta pora	Vyta pora	Koaksialinis kabelis, 4-laidų vyta pora
signalų charakteristikos	loginis „1“	0V	>=80%VDD	<=0.3VDD	+3V	<=25%VDD	<=10%VDD
	loginis „0“	2V	<=20%VDD	>=0.7VDD	-3V	>=75%VDD	>=90VDD
	kita		VDD=8..18V	VDD=5V ~TTL	VDD=5..24V	VDD= –7..12 V	VDD=2.5...3.3V
galvaninis atskyrimas yra/nėra		Gali būti išorinis	Gali būti išorinis	Gali būti išorinis - gana problematiška ir neapsimoka	Gali būti išorinis	Gali būti išorinis	yra
palaikomas mazgų skaičius		256	15	112	64	32	Virtualiai neribojamas
ar yra aparatinis klaidos atstatymas		yra	nėra	Nėra	nėra	yra	nėra
ar yra klaidų aptikimas		yra	nėra	Nėra (yra tik sinchronizacijai)	nėra	yra	Nėra
ar egzistuoja integruotas protokolas	yra/nėra	Yra (atsiliepia kainai, dažnai reikalingi išoriniai komponentai)	Yra (kainos beveik neįtakoja)	Yra (kainos beveik neįtakoja)	Yra (retas, atsiliepia kainai)	Yra (atsiliepia kainai)	Yra (reikalingi išoriniai komponentai; greitas MV, atsiliepia kainai)
	mv pvz	C8051F040, 87C196CB	ATA6602, TA6603	MPC500, MC68300	LM3S1000, LM3S8000	DPC31	PIC18F97J60, DS80C400
koks naudingų duomenų santykis su visu persiųstų duomenų kiekiu (pateiktas apytikslis santykis)		80/106 ~ 0,74	76/88 ~ 0,86	16/29 ~ 0,55	2024/2056 ~ 0,98	64/128 ~ 0,5	100/126 ~ 0,79
ryšio linijos režimas		Dvikrypčiai duomenų mainai	Dvikrypčiai duomenų mainai	Dvikrypčiai duomenų mainai.	Pusiau dvikrypčiai	dupleksinis, pusiau dupleksinis	dvikrypčiai
sinchronizacijos būdas tarp mazgų		savi sinchronizacija	savi sinchronizacija	SCL linija skirta sinchronizacijai	savisinchronizacija	Sinchroninis/asinchroninis	Sinchroninis/asinchroninis

VDD – maitinimo įtampa;

Iš pateiktos lentelės (3 lentelė) matyti, kad pasirinktos nagrinėti sąsajos skiriasi savo savybėmis. Iš analizuotų sąsajų galima pasakyti, kad sąsajos PROFIBUS-DP, ETHERNET gali būti naudojamos MV tinklo sujungimui su kitu MV tinklu. Jos pasižymi dideliu pralaidumu bei palaiko didelį atstumą tarp mazgų, tačiau šios sąsajos neturi klaidų aptikimo (klaidų aptikimas gali būti realizuotas programiškai) ir integruotas sąsajų palaikymas mikroprocesoriniuose valdikliuose yra retas. PROFIBUS-DP ir ETHERNET sąsajų realizacija, nesant integruoto palaikymo mikroprocesoriniame valdiklyje, naudoja daug MV resursų.

Sąsają LIN galima panaudoti jutikliams, vykdymo įrenginiams sujungti į tinklą, bet ir tai tik tiems, kuriems reikalingi maži duomenų srautai bei mazgų kiekis yra gana mažas. Taip pat LIN sąsajos galvaninis atrišimas yra sudėtingas. Sąsaja LIN netinka MV sujungimui į tinklą, o sąsajos MODBUS, I2C, CAN kaip tik gali būti naudojamos MV į tinklą sujungti.

MODBUS sąsaja pasižymi pakankamai dideliu greičiu, taip pat galima prijungti didelį mazgų skaičių, tačiau neturi aparatinio klaidų aptikimo ir atstatymo, o galvaninis atrišimas gali būti tik išorinis.

I2C sąsaja pasižymi sąlyginai mažu greičiu, galima sujungti daug mazgų, bet sąsajoje mazgai negali būti labai nutolę vienas nuo kito. Taip pat sąsaja nėra atspari trikdžiams, neegzistuoja aparatinis klaidų aptikimas ir jų atstatymas.

CAN sąsaja pasižymi geromis savybėmis: dideliu greičiu, mazgai gali būti daug nutolę viens nuo kito, sąsajoje gali būti daug mazgų, ji atspari trikdžiams, egzistuoja aparatinis klaidų aptikimas ir atstatymas, nemažai valdiklių turi integruotą sąsajos palaikymą. Tiesa, sąsajos realizacija yra santykinai brangi.

Palyginus sąsajas matyti, jog optimaliausia sąsaja mikroprocesorinių valdiklių sujungimui yra CAN sąsaja.

1.2 CAN sąsajos protokolai

CAN sąsajos specifikacijos 2.0A ir 2.0B aprašo OSI modelio fizinį ir ryšio kanalinį lygmenis [CBB05+]. Šiais laikais šie du lygmenys yra palaikomi aparatinėmis įrangos priemonėmis.

Siekiant sujungti skirtingų gamintojų įrenginius pasižyminčius skirtingomis techninėmis charakteristikomis bei norint standartizuoti duomenų perdavimą, įrenginių identifikavimą, adresavimą, incializacijos procedūras, perduoti didesnius informacijos kiekius didesnius nei 8 baitai, reikalingas aukštesnio lygio aprašymas nei fizinis ar ryšio kanalinis. CAN sąsajai naudojami 3 iš 7 OSI modelio lygmenys: fizinis, ryšio kanalinis ir taikomasis. CAN sąsajai

sukurta nemažai taikomojo lygmens protokolų. Šių protokolų įvairovė yra ganėtinai įvairi, bet populiariausi ir daugiausiai naudojami yra šie: CANOpen, DeviceNet, CAN Kingdom. Protokolai dažniausiai naudoja OSI modelį, bet yra ir tokių, kurie taiko kitokį modelį. Tokio protokolo pavyzdys gali būti CAN Kingdom. Toliau panagrinėsiu išvardintus protokolus bei palyginsiu juos tarpusavyje.

1.2.1 CANopen

Tai CAN sąsaja paremtas bei standartizuotas, lanksčiai konfigūruojamas protokolas. Šis protokolas buvo suprojektuotas lengvųjų automobilių pramonei, tačiau išplito ir dabar taikomas labai įvairiose srityse, tokiose kaip kavos aparatai ar medicininė įranga.

Šiame protokole kiekvienas mazgas laikomas objektu ir turi turėti objekto žodyną (OD), kuriuo aprašoma ką objektas gali daryti ir kokiomis mazgo galimybėmis gali naudotis kitas tinklo mazgas.

CanOpen protokolo taikomajame lygmenyje pranešimai perduodami susisiekimo objekto COB viduje. COB objektas tai CAN sąsajos pranešimas. Kiekvienas COB objektas sudarytas iš duomenų iki 8 baitų ir unikalaus identifikatoriaus COBID, kuris apibrėžia mazgo adresą bei funkcijos kodą. COBID nurodo, kokio tipo yra COB objektas:

- duomenų apdorojimo objektas (Process Data Object PDO);
- duomenų serviso objektas (Service Data Object SDO);
- administravimo objektas;
- specialios paskirties objektas.

PDO ir SDO yra susisiekimo objektų tipai, kurie tarp mazgų perduoda taikomuosius duomenis.

Duomenų serviso objektai SDO naudoja kelis CAN pranešimus duomenų, didesnių nei 8 baitai, pedavimui. Taip pat SDO objektų pagalba galima iškviesti kitų mazgų teikiamas funkcijas, kurios aprašytos kiekvieno mazgo objekto žodyne. Šie objektai naudojami didelių informacijos kiekių perdavimui, tačiau šių objektų organizuojami duomenų perdavimai yra žemo prioriteto. SDO objektai naudoja trijų tipų duomenų perdavimo metodus:

1. Paspartintas perdavimas – naudojamas mažiems duomenų kiekiams perduoti. Šis perdavimas nereikalauja jokios išanksinės inicializacijos. Šiuo perdavimu perduodama ne daugiau nei 4 baitai informacijos, todėl jie siunčiami tame pačiame CAN pranešimo duomenų lauke. Pirma CAN pranešimo 8 baitų pusė panaudota perdavimo informacijai talpinti, pavyzdžiui: kur saugoti siunčiamus duomenis ar kiek baitų iš siunčiamų

duomenų nėra duomenys. Antrą 8 baitų pusę sudaro perduodami duomenys. Kiekvienas iš CANopen mazgų privalo palaikyti paspartintą perdavimą.

2. Segmentuotas perdavimas – naudojamas nuo 4 iki 128 baitų perdavimui. Šis perdavimas naudoja COB objektą serverio mazgo inicializavimui, praktiškai identišką paspartinto perdavimo objektui.. Šis objektas paruošia serverio mazgą ir laukia serverio mazgo atsako klientui: ar numatomas dialogas gali būti užmegztas ar ne. Po inicializacijos duomenys perduodami CAN pranešimais, kuriuose vienas CAN pranešimo duomenų lauko baitas, nurodo kelintas segmentas perduodamas, o sekantys 7 baitai yra perduodami duomenys. Po kiekvieno duomenų segmento, gavėjas, jei pripažįsta segmentą, siunčia atsakymą siuntėjui. Sistemos mazgams galima laisvai parinkti ar palaikomas segmentuotas perdavimas.
3. Blokinis perdavimas – naudojamas neribotam duomenų kiekiui perduoti. Šio perdavimo inicializavimas ir duomenų perdavimas panašus į segmentuotą perdavimą. Pagrindinis skirtumas yra tas, kad siunčiami duomenys padalinami į segmentus ir siunčiant segmentus patvirtinimai iš gavėjo siunčiami tik paskutiniam segmento pranešimui.

Duomenų apdorojimo objektai PDO – naudojami nedideliems duomenų kiekiams iki 8 baitų perduoti. Šių objektų pranašumas - aukštesnis siuntimo prioritetas. PDO objektai turi būti apibrėžti tiek kliento, tiek serverio mazguose naudojant PDO.

Administravimo objektai – atsakingi už tinklo valdymą bei identifikatoriaus teikimo paslaugas.

Specialios paskirties objektai – tai sinchronizavimo, laiko žymės, klaidos, stebėjimo objektai

Šio protokolo privalumai: plačiai naudojamas Europoje; naudojamas daugelyje skirtingų sričių: pramonėje, traukinių ir medicinos įrangoje; palaikomas maksimalus CAN sąsajos greitis.

Minusai: labai sudėtinga specifikacija; nepalaiko išplėsto CAN sąsajos pranešimo; nėra apsaugos nuo neteisingo mazgo greičio nustatymo; mazgo identifikacijai naudojami 7 bitai iš 11 CAN pranešimo identifikatoriaus, kas sumažina galimų prijungti mazgų kiekį iki 128. Platinamos protokolo realizacijos ir konfigūravimo įrankiai brangūs.

1.2.2 DeviceNet

DeviceNet atviras CAN sąsajos standartas 1990 metais sukurtas Allen Bradley [RL03]. Norėdamas pasiekti, kad šis protokolas būtų plačiai naudojamas ir paplitęs pasaulyje, Allen Bradley perdavė protokolo plėtojimą ODVA (Open DeviceNet Vendor Association) asocijacijai.

DeviceNet protokolas susisiekimo paslaugų, duomenų ir elgesio apibūdinimui naudoja abstrakčius objektų modelius. Šie objektai organizuoja sistemos elgseną ir grupuoja duomenis, naudodami virtualius objektus. Kiekvienas mazgas gali turėti daugialypius tapatybės objektus. Objektai teikia tokias paslaugas kaip atributo paėmimas (get atribute) ar atributo priskyrimas (set atribute). Visuose mazguose dalyvauja tapatybės ir DeviceNet objektai.

Tapatybės objektą sudaro tokie duomenys: pardavėjo numeris, prietaiso tipas, serijinis numeris.

DeviceNet objektas apibūdina ryšį ir jį sudaro tokie parametrai: mazgo adresas ir tinklo sparta.

DeviceNet naudoja 2.0A CAN sąsajos specifikaciją, taigi šis standartas apibrėžia 11 bitų pranešimo identifikatorius. Pranešimai skirstomi į keturias grupes. Šios grupės apibrėžia bendravimo struktūrą, tarp bet kokių dviejų mazgų. Grupėms priskiriami prioritetai naudojant CAN pranešimo identifikatoriaus bitus. Grupė „1“ turi aukščiausią prioritetą.

DeviceNet naudoja dvi pranešimų schemas: tikslaus pranešimo (Explicit Messaging) ir įvesties/išvesties pranešimo (I/O). Tikslūs pranešimai konfigūruoja mazgus ir užtikrina bendrą susisiekimą tarp bet kokių dviejų mazgų. Pranešimų valdymui naudojama dalis CAN sąsajos pranešimo duomenų lauko. Įvesties/išvesties pranešimai naudojami duomenų perdavimui. Šioje pranešimų schemoje visas CAN sąsajos pranešimo duomenų laukas naudojamas duomenims talpinti. Perduodami duomenys mazguose aprašomi elektroninėse duomenų lentelėse (EDS). Šios lentelės patogios, nes vedantieji mazgai gali nuskaityti kiekvieno mazgo EDS bei tą mazgą sukonfigūruoti .

Šio protokolo pliusai: labai daug įrenginių,įdiegtų su šiuo protokolu; ODVA efektyviai platina šią technologiją.

Minusas- palaikomi tik 64 mazgai.

1.2.3 CAN Kingdom

Tai vienas iš išskirtinių CAN sąsajos protokolų. Kitaip nei kiti CAN sąsajos protokolai, šis protokolas nenaudoja OSI modelio. Šis protokolas elgesio organizavimui naudoja tokį modelį:

valstybės pašto sistema su miestais, merais ir laiškais. Šiame modelyje valdantysis įrenginys vadinamas – Sostine. Programinė įranga kontroliuojanti sostinę vadinama – Karaliumi. CAN sąsaja ir CAN protokolas vadinamas pašto sistema. CAN sąsajos mazgas vadinamas – Miestu, kuris valdomas mero (taikomojo kodo). Kiekvienas miestas turi CAN kontrolerį – pašto skyrių, kuris valdomas pašo viršininko (kodo), teikiančio tam tikras paslaugas. Duomenų perdavimai vykdomi laiškais, kurie siunčiami visiems mazgams. Laiškas yra vienas CAN sąsajos pranešimas. Laišką sudaro lapas įdėtas į voką (CAN identifikatorius). Lapą sudaro 8 eilutės (CAN sąsajos pranešimo duomenų lauko 8 baitai). Jei reikia daugiau nei 8 eilučių, perdavimui jos įdedamos į dokumentą, kuris sudarytas iš vieno ar daugiau lapų, tačiau su tuo pačiu voku.

CAN sąsajos mazgai CAN Kingdom protokole neturi iš anksto nustatytų adresų. Pranešimų identifikatoriai, tai CAN sąsajos identifikatoriai. Daugelis mazgų gali naudoti tą patį CAN sąsajos identifikatorių, išskyrus identifikatorių 00000000. Karalius (vedantysis) sau rezervuoja identifikatorių 00000000. CAN mazgai iššifruoja taisyklių rinkinius, kurie nurodo kaip užšifruoti ir iššifruoti duomenis. CAN Kingdom protokolo esmė - vedantis įrenginys. Vedantysis įrenginys visiems mazgams perduoda taisyklių rinkinį, t.y kaip bus perduodama informacija iš vieno mazgo į kitą. CAN Kingdom protokolo vedantysis sukuria CAN protokolo primityvus CAN sąsajos mazgams. Sistemą projektuojantis formas naudoja mazgų kontrolei.

Protokolo privalumai: CAN mazgo architektūra visiškai atskirta nuo sistemos architektūros, projektuotojui nereikia rūpintis mazgo identifikatoriumi; formos gali būti naudojamos mazgo nustatymui (setup); kartu su šiuo protokolu vienoje sistemoje gali būti įdiegti kiti protokolai; galima prijungti iki 256 įrenginių; leidžiamas maksimalus CAN sąsajos greitis.

Minusas - paini šio protokolo terminologija.

1.2.4 Protokolų palyginimas

Apžvelgus protokolus galima daryti išvadą, kad ir kokie skirtingi jie visi būtų, CAN sąsajos taikomojo lygmens protokolai sprendžia panašias problemas, tokias kaip identifikatoriaus paskirstymas, informacijos, didesnės nei 8 baitai, perdavimas ir kt. Protokolo pasirinkimą lemia tokie veiksniai kaip: kaina, protokolo palaikymas, dokumentacija ir kt [Ti04]. Visų apžvelgtų protokolų specifikacijos yra laisvai prieinamos, tačiau konfigūravimo įrankiai CANopen ir DeviceNet protokolams yra ganėtinai brangūs. Todėl norint realizuoti sistemą su minimaliomis piniginėmis sąnaudomis šių protokolų pasirinkti negalima. Taip pat kiekvienas iš šių protokolų turi turėti bent vieną vedantyjį įrenginį, kas riboja paskirstytos sistemos galimybes. T.y. sugedus

vedančiajam įrenginiui sistema išeina iš rikiuotės. Taigi lanksčiausias ir patogiausias protokolas, kuris gali būti pritaikytas savo reikmėms yra CAN Kingdom protokolas. Pasinaudodamas CAN Kingdom protokolo principais suprogramuosiu paskirstytą mikroprocesorinių valdiklių sistemą be vedančiojo įrenginio.

2. Mikroprocesoriniai valdikliai

Mikroprocesoriai valdikliai tai esminis pramonės automatizavimo veiksnys. Juos naudojant galima lanksčiai prisitaikyti prie besikeičiančių procesų, o taip pat greitai nustatyti ir eliminuoti sutrikimus bei klaidas. Mikroprocesorinis valdiklis – tai naujas terminas, kurį imta naudoti 70-aisiais metais. Mikroprocesorinis valdiklis – tai programiniu būdu valdomas skaitmeninis informacijos apdorojimo prietaisas, sudarytas iš vieno arba kelių didelių integrinių grandynų. [DJ00] Mikroprocesorinį valdiklį sudaro: operacinis įtaisas, valdantysis įtaisas (suformuojantis valdančiuosius signalus operaciniam įtaisui), programinė ir duomenų atmintis, periferiniai įrenginiai bei įranga, skirta išorinių įtaisų prijungimui. Tai integrinis grandynas su specializuotu mikroprocesoriumi, atitinkantis šiuos kriterijus: jis pritaikytas darbui paprasčiausiuose įtaisuose, suprojektuotas taip, kad jį būtų galima naudoti matavimo-valdymo sistemose, todėl jis turi skaitmeninius ir analoginius prievadus darbui su įvestimi/išvestimi. MV gali būti dviejų architektūrų: CISC (Complex Instruction Set Computer) - kompiuteris su pilnu komandų rinkiniu ir RISC (Reduced Instruction Set Computer) - kompiuteris su sumažintu komandų rinkiniu. Mikroprocesoriniai valdikliai gali būti sudaryti iš daugelio įrenginių. Tipinio MV sudėtis:

- Procesorius;
- Taktinių impulsų generatorius;
- Programų atmintis;
- Duomenų atmintis;
- Pertraukimų sistema;
- DMA(Direct Memory Access)
- Keitikliai: analogas – kodas, skaičius – analogas;
- Komparatoriai;
- Laikmačiai;
- Skaitikliai;
- Prievadai.

Prie mikroprocesorinių valdiklių galima prijungti papildomus įrenginius pagal atliekamo uždavinio specifiką pvz:

- Jutiklius;
- Indikatorius;
- vykdymo grandines;
- Ir kt.

3. Mikroprocesorinių valdiklių sistemos

Šiame skyriuje apžvelgsiu centralizuotų ir paskirstytų sistemų skirtumus. Taip pat realaus laiko mikroprocesorinių valdiklių sistemas.

3.1 Paskirstytų ir centralizuotų sistemų skirtumai

Šiuo metu pastebimas perėjimas nuo centralizuotų sistemų, kuriose vienas galingas procesorius valdo visus periferinius įrenginius, prie paskirstytų – kada kiekvienas sistemos elementas tampa aktyviu mazgu. Kaip žinoma centralizuotų sistemų privalumai yra tokie: paprastesnis sistemos valdymas, mažiau lėšų reikia išleisti aptarnaujančiam personalui, pakanka vieno kompiuterio kelioms, tos pačios firmos, informacinėms sistemoms aptarnauti. Tačiau centralizuotas sistemas sudėtinga plėsti bei sugedus sistemos procesoriui sutrinka visas sistemos darbas. Šiuos trūkumus eliminuoja paskirstytos sistemos. Kompiuteriai pasaulyje dabar gali susisiekti vienas su kitu kompiuterių tinklais. Tuomet atsiranda galimybė vieno kompiuterio vartotojui naudotis kito kompiuterio resursais (bylomis, programomis, spausdintuvais ir kt.). Paskirstyto duomenų apdorojimo sistema yra priešingybė centralizuotai sistemai. Ją sudaro išbarstyti įrenginiai, kurių kiekvienas gali savarankiškai atlikti įvairias duomenų apdorojimo operacijas, vykdyti įrenginiui paskirtas užduotis. Atskiri paskirstytų sistemų įrenginiai dažniausiai specializuojami tam tikros klasės uždaviniams spręsti. Pavyzdžiui, įrenginiai gali būti skiriami temperatūros matavimui, variklių darbui užtikrinti.

Paskirstytų sistemų privalumai:

- *Paskirstyti įrenginiai*. Sistemą sudaro keletas arba daug ryšio kanalais sujungtų įrenginių, kurie gali būti įvairiose vietose bei įrenginių kiekis gali būti lengvai padidintas prijungus papildomus įrenginius;
- *Paskirstytas duomenų apdorojimas*. Sistemos uždavinių sprendimui galima panaudoti iš karto kelis arba net visus sistemos įrenginius, todėl sistemos uždaviniai vykdomi ypač sparčiai;
- *Paskirstyti duomenys*. Sistemos tvarkomi duomenys gali būti paskirstomi saugojimui keliuose sistemos įrenginiuose taip, kad juos visus kartu būtų galima naudoti sistemos sprendžiamuose uždaviniuose.
- *Paskirstytas valdymas*. Paskirstytos sistemos paprastai nėra valdomos ir tvarkomos centralizuotai. Bet kartais paskirstytas valdymas yra derinamas su centralizuotu. Pavyzdžiui, didelėse sistemose pageidautina centralizuotai sekti viso įrenginių tinklo būklę.

- Funkcijų rezervavimas.

Paskirstytų sistemų trūkumai:

- Sunkesnis gedimų diagnozavimas;
- Sudėtingas bendrų informacinių resursų valdymas

Paskirstytos sistemos tikslas - realizuoti kompleksinius valdymo uždavinius. Jei valdymo uždavinio apimtis viršija atskiro prietaiso galimybes arba, dėl struktūrinių ar technologinių priežasčių, tikslinga valdymo funkcijas decentralizuoti, valdymo uždavinys paskirstomas į keletą modulių.

Skirstant valdymo uždavinius galimos dviejų tipų struktūros: plokštumos struktūra ir hierarchinė struktūra. Su mikroprocesoriniais valdikliais, realizuotose paskirstytos sistemose, labiau paplitusi hierarchinė struktūra, bet praktikoje dažnai sutinkamos abiejų tipų struktūros. Projektuojant paskirstytą sistemą, projekto skirstymas į modulius ir jų funkcijų aprašymas yra viena svarbiausių valdymo algoritmo projektavimo dedamųjų. Svarbu yra aprašyti modulių duomenis ir funkcijas bei nustatyti jų vykdymo rėžimus ir eiliškumą. Modulio egzemplioriai, laiko valdymo funkcijomis, priskiriami atitinkamoms užduotims. Modulių aprašyme būna ir komunikacijų keliai, t.y. duomenų mainuose dalyvaujantys kintamieji. Iki šios projektavimo stadijos valdymo algoritmas kuriamas neapsiribojant konkrečiais prietaisais. Vėlesniame projektavimo etape, naudojantis specifinėmis prietaisų valdymo funkcijomis, sistemos moduliai instaliuojami konkrečiuose prietaisuose, t.y. nustatoma sistemos konfigūracija. Konfigūruojant sistemą taip pat nustatoma mazgų duomenų mainų apimtis, mazgo kaip tinklo elemento tipas valdantysis (master) ar valdomasis (slave).

3.2 Realaus laiko sistemos

Realaus laiko sistemos tai sistemos, kurios turi ne vien įvykdyti užduotį, bet ir įvykdyti užduotį užsibrėžtu laiku. Užduoties įvykdymo laikas, realaus laiko sistemose, laibai svarbus (pvz. orlaivių eismo valdymas, robotų valdymo sistema fabrike). Realaus laiko sistema turi būti prognozuojama net tik loginiu, bet ir laiko požiūriu. Sistemą galima vadinti realaus laiko sistema, jei ji atitinka šiuos reikalavimus: reakcija turi būti ne ilgesnė, negu nustatyta trukmė; jei darbas multiprograminis, tuomet sistema turi laikytis griežto tvarkaraščio; reikia valdyti procesų prioritetus; sistema turi palaikyti prognozuojamus procesų komunikacijos metodus. Realaus laiko sistemose turi būti žinomi pertraukimų lygiai (dažniausiai sistema valdoma pertraukimais, kurių prioritetai ir nusako užduoties svarbą).

Užduotys, realaus laiko sistemose, skirstomos į:

- periodines – aktyvuojamos periodiškai, kas tam tikrą laiko tarpą T.
- aperiodinės – užduoties aktyvavimas nenuspėjamas, gali įvykti bet kuriuo laiko momentu;
- pavienės(sporadic) – kaip ir aperiodinės užduotys, tik pridedami papildomi reikalavimai aktyvavimui t.y. gali aktyvuotis ne dažniau kaip tam tikrą laiką.

3.3 Realaus laiko operacinės sistemos

Sistemos naudojančios MV pasižymi tokiomis savybėmis: ribotu procesoriaus galingumu (1-100 Mips), ribota programų atmintmi (1-512 KB), ribota operatyviąja atmintimi (64-16384 baitų, beveik niekada neviršija 64KB) [Das08]. Tokios savybės kelia ypatingus reikalavimus sistemos programinei įrangai. Pagal pateiktas savybes galimos kelios programinės įrangos realizacijos: pagrindinis ciklas su pertraukimų metodais, realaus laiko operacinės sistemos (RTOS).

Pagrindinis ciklas su pertraukimų metodais. Paralelizmas pasiekiamas pertraukimais, kuriuos apdorojus grįžtama į pagrindinį ciklą. Tokios realizacijos minusai yra tokie:

- naudojami globalūs kintamieji;
- sunku išskaidyti programą į atskirus modulius;
- sudėtingesnis programavimas ir programos tobulinimas.

O tokios realizacijos privalumai yra šie:

- gali būti parinktas optimalus realaus laiko įvykių aptarnavimo scenarijus, todėl reakcijų į įvykius trukmės gali būti minimalios;
- mažesni reikalavimai sistemos resursams (atminties dydžiui; procesoriaus greitaeigiškumui), todėl mažesnė sunaudojama galia ir aparatinės dalies kaina.

Realaus laiko operacinė sistema. Operacinė sistema teikia patogius ir lengvai pritaikomus programavimo šablonus taikomosios programinės įrangos kūrimui [Hri07]. Operacinės sistemos pagalba valdomas užduočių skirstymas bei abstrahuojamas aparatinės įrangos valdymas (programuotojo izoliacija nuo aparatinės įrangos detalių). Norint operacinę sistemą pavadinti realaus laiko sistema, ši sistema turi turėti tokius požymius: ji privalo būti deterministinė ir garantuoti blogiausio atvejo pertraukties vėlinimo bei užduočių perjungimo laikus [Bar03].

Šiuo metu sukurta ganėtinai daug realaus laiko operacinių sistemų (RTOS), kurios skirtos mikroprocesorinių valdiklių valdymui. 4 lentelėje pateikiu laisvai platinamų ir komercinių RTOS sąrašą. Šis sąrašas yra nebaigtinis, nes pasirinktos tokios RLOS, kurios galėtų dirbti su AVR mikroprocesoriniais valdikliais.

4 lentelė laisvai platinamų ir komercinių RTOS *sąrašas.*

Laisvai platinamos RTOS		
RLOS pavadinimas	**Kilmė**	**Internetinis puslapis**
TinyOs	Berklio universitetas	www.tinyos.net
Contiki	Švedijos kompiuterijos institutas	www.sics.se/contiki
Mantis	Kolorado universitetas	http://mantis.cs.colorado.edu
Timber	Lulea technologijų universitetas	www.ltu.se/
Nano-RK	Carnegie Mellon universitetas	www.nanork.org
FreeRTOS	John Westmoreland	www.freertos.org
Komercinės RTOS		
RLOS pavadinimas	**Gaminojas**	**Internetinis puslapis**
Velocity	Green Hills	www.hgs.com
MicroC/OSII	Micrium	www.micrium.com
ThreadX	Express logic Inc.	www.rtos.com
CMXTiny+	CMX systems	www.cmx.com
µnOS	Miray Software	www.mirray.de

Realaus laiko operacines sistemas tikslinga naudoti kai mikroprocesorinio valdiklio galimybės išnaudojamos nepilnai. Tačiau šiame darbe projektuojama sistema dirbs arti savo aparatinių galimybių ribos, todėl RTOS taikymas nėra tikslinngas.

4. Paskirstytos matavimo sistemos projektavimas

4.1 Sistemos komponentų pasirinkimas

Išanalizavus pasirinktas technologijas, iš jų pasirinktos tinkamiausios. Renkantis sistemos komponentus pirmiausia reikia išsirinkti sąsają, nes sąsajų pasirinkimas yra ganėtinai ribotas t.y. galima rinktis iš kelių dešimčių sąsajų, o MV šeimų, ir ypač tų šeimų realizacijų, egzistuoja žymiai daugiau. Renkantis atvirkščiai išaugtų sistemos kaina, nes sąsajos palaikymui reiktų naudoti papildomus įrenginius.

4.1.1 Sąsajos pasirinkimas

Išnagrinėjus nuosekliąsias sąsajas (CAN, LIN, I2C, MODBUS, PROFIBUS-DP), kurios skirtos mikroprocesorinių valdiklių sujungimui į tinklą, galima daryti išvadą, kad optimaliausia pasirinkti CAN (controller area network) sąsaja. Pasirinktos CAN sąsajos pagrindinės techninės charakteristikos yra tokios:

- greitis iki 1Mbit/s;
- atstumas iki 1000m;
- įrenginių kiekis vienam segmentui iki 256.

4.1.2 Mikroprocesorinio valdiklio pasirinkimas

Šiais laikais MV gamina daugelis kompanijų, o kaip lyderius galima būtų paminėti: Freescale(buvusi Motorola), Renesas (buvusi Hitachi), NXP (buvusi Philips semiconductor), NEC, Atmel, Microchip, ST, Infineon (buvusi Siemens), Texas Instruments, Zilog, Analog Devices, Dallas Semiconductor, ir daugelį kitų. Tiek MV gamintojų tiek ir pačių MV pasirinkimas yra labai platus.

Sistemos projektavimui pasirinkti Atmel kompanijos mikroprocesoriniai valdikliai AT90CAN128. Tai 8 bitų RISC mikroprocesorinis valdiklis, kuris yra ganėtinai paprastas bei lengvai konfigūruojamas lyginant su 16 ar 32 bitų MV. Šio MV architektūra, AVR, tai viena efektyviausių, šiuo metu rinkoje esančių, 8 bitų architektūrų. Šį MV AT90CAN128 galima būtų lyginti su H8 (Renesas), MSP430 (Texas Instruments), ARM7(Atmel, ST, NXP, TI...) panašaus kainų lygio MV ir išskirti tokius skirtumus:

- H8 (Renesas) 8 bit: sunkiai prieinamos kūrimo priemonės, didesnė kaina;
- MSP430 (Texas Instruments) 16 bit: pigiausias, ekonomiškiausias (paskutiniai ARM Cortex jau jį aplenkė), bet nėra integruotas CAN sąsajos palaikymas;

- ARM7, Cortex 32 bit: Sudėtingesnis programavimas, montažas.

Taip pat šis MV pasirinktas todėl, kad jame yra visiškai integruota CAN sąsaja, atitinkanti 2.0 A ir B sąsajos standartus. AT90CAN128 mikroprocesorinis valdiklis pasižymi šiomis savybėmis: mažomis energijos sąnaudomis (dirba nuo 2.7 V iki 5.5 V), maksimaliu darbiniu dažniu: 8 MHz, kai įtampa 2.7 V ir 16 MHz, kai įtampa 5.5 V (vykdoma viena instrukcija per vieną taktą), 32 (8 bitų) pagrindiniais registrais, periferiniai kontrolės registrais, 128Kb perprogramuojama flash tipo programų atmintimi, 4 Kb vidine dinamine RAM atmintimi, 8-bitų sinchroniniu laikmačiu (angl. timer), 8-bitų asinchroniniu laikmačiu (angl. timer), 16-bitų sinchroniniu laikmačiu (angl. timer), 53 įvesties/išvesties perprogramuojamomis linijomis, QFN korpusu [Atm07]. Šio valdiklio naudojamos galios santykis su skaičiavimo sparta nėra didelis – 1.5 – 2 mW/Mhz ir stipriai atsilieka nuo šiuo metu pagal šį parametrą pirmaujančių ARM (32 bitų) bei MSP430 (16 bitų) valdiklių 0.06(dėl 0.13 μm technologijos, kuri neturi analogų tarp AVR valdiklių) – 0.2 – 0.6 mW/Mhz. Palyginimas neatsižvelgiant į skiltiškumą yra įmanomas, nes daugiausia atliekami valdymo ir duomenų siuntimo uždaviniai, kur didesnis apdirbamų duomenų skiltiškumas nėra ryškus privalumas.

4.2 Sistemos funkciniai reikalavimai

Kaip žinoma, šiais laikais paskirstytos sistemos sparčiai vystosi ir tuo pačiu užima vis didesnę automatizuotų sistemų dalį. Automatizuotų sistemų pagrindinis privalumas, lyginant su įprastinėmis sistemomis, kuriose dalyvauja žmogus, yra ekonominio pobūdžio. Šios sistemos suteikia galimybę gerokai sumažinti aptarnaujančio personalo skaičių, operatyviau reguliuoti suvartojamos energijos kiekį, tiksliau numatyti energijos poreikius bei sumažinti išlaidas. Tai ypač patogu didelėse įmonėse, kuriose darbo procesas nenutrūksta visą parą arba reikalingas preciziškas tikslumas bei svarbu išvengti klaidų.

Šiame darbe realizavimui pasirinkta paskirstyta matavimo sistema. Paskirstyta mikroprocesorinių valdiklių sistema realizuoja kompleksinius valdymo uždavinius, kurie būna įvairaus pobūdžio. Paskirstytos sistemos pranašesnės už centralizuotas tuo, kad kiekvienas jų mazgas veikia nepriklausomai bei vieno ar kelių mazgų sutrikimas nesustabdo viso darbo proceso, tačiau įtakoja darbo kokybę ir greitį.

Paskirstyta mikroprocesorinių valdiklių sistema susideda iš tokių dalių:

1. Mazgų, kurie gali būti sudaryti iš mikroprocesorinių valdiklių bei prie jų prijungtų įvairių periferinių įrenginių;
2. Ryšio sąsajos.

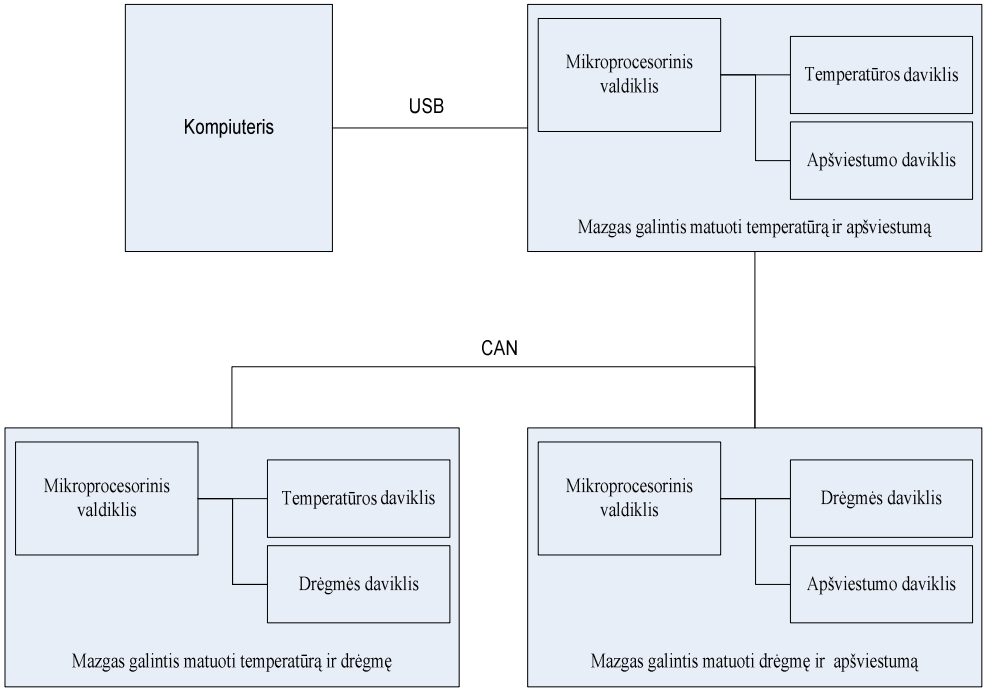
Projektuojamos sistemos mazgai gali matuoti atskirus fizikinius dydžius: temperatūrą, drėgmę, apšviestumą. Kiekvienam iš mazgų priskiriama matuoti vieną arba kelis tam tikrus dydžius. Kiekvienas sistemos mazgas turės perteklinį funkcijų kiekį t.y. mazgai atsakingi už matavimus galės matuoti temperatūrą, drėgmę ir apšviestumą vienu metu. Šiomis perteklinėmis funkcijomis galima bus pasinaudoti tada, kai vienas ar keli sistemos mazgai išeis iš rikiuotės. Taigi jei vienas ar keli mazgai nedirbs, sistema vistiek galės atlikti jai paskirtą darbą, tačiau su kokybiniais nuostoliais.

4.3 Sistemos blokinė schema

Projektuojamos paskirstytos matavimo sistemą sudarančius tris mazgus galima apibūdinti taip:

1. Mazgas galintis matuoti temperatūrą ir apšviestumą. Šis mazgas bus atsakingas ir už duomenų surinkimą bei perdavimą į kompiuterį per USB sąsają
2. Mazgas galintis matuoti temperatūrą ir drėgmę.
3. Mazgas, matuojuosiantis temperatūrą ir drėgmę. Tai matavimo mazgas, kuris reikalui esant gali papildomai matuoti ir apšviestumą

Sistemos blokinė schema pateikta 8 pav. Tokia sistema gali būti pritaikyta šiltnamyje, o ją modifikavus ji gali būtų pritaikyta namo šildymo sistemoje ar eksperimento automatizavime. Kaip matoma sistemos blokinėje schemoje, sistemą lengva praplėsti t.y. pridėti daugiau sistemos mazgų, taip išplėčiant sistemos bendrą funkcionalumą.



8 pav. Sistemos blokinė schema.

4.4 Sistemos principinė schema

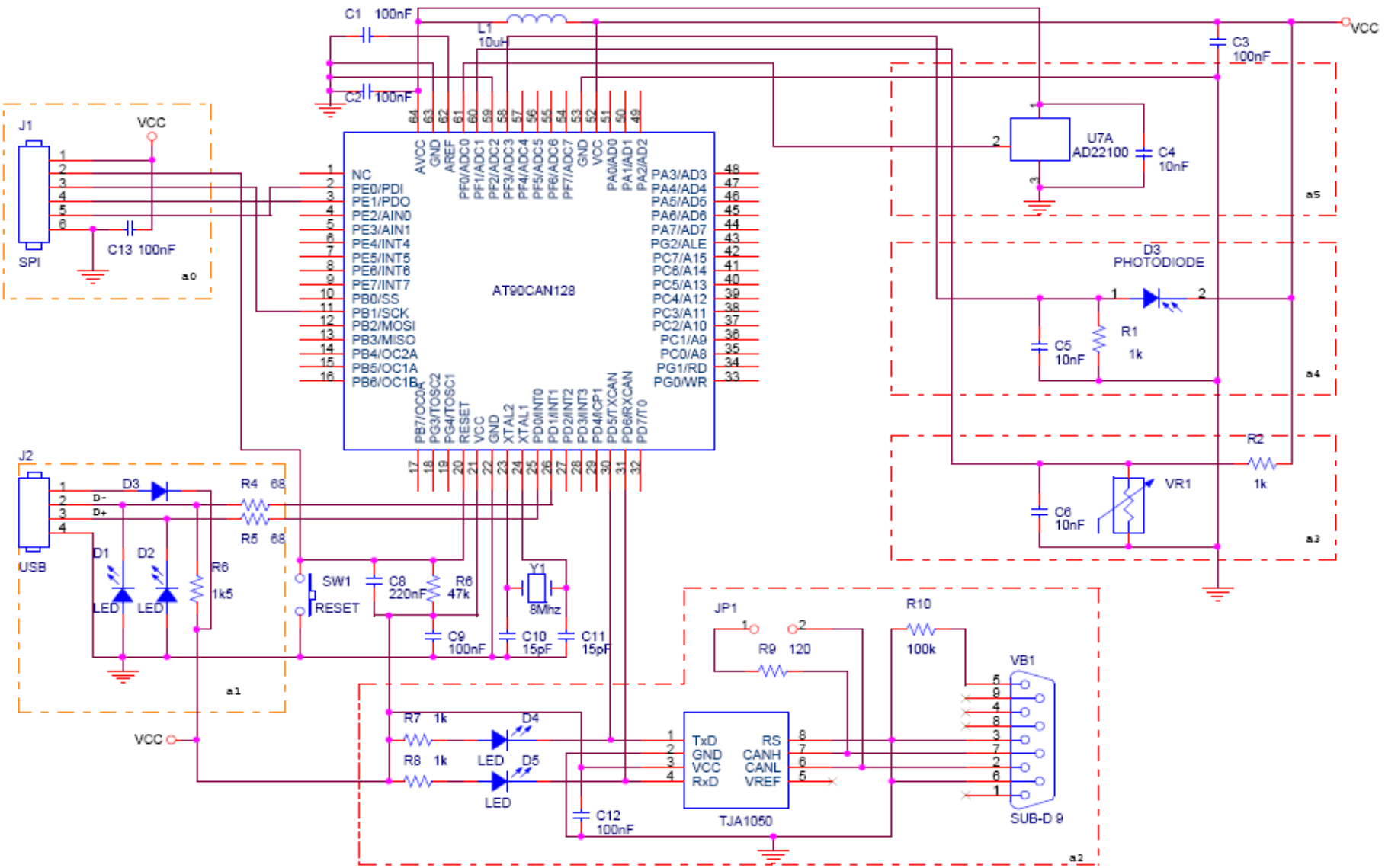
Pasirinkus mikroprocesorinius valdiklius ir sąsają, kuria bus sujungti sistemos mazgai, bei sudarius sistemos blokinę schemą galima projektuoti sistemos principinę schemą.

Principinės schemos projektavimui pasirinkta Orcad 9.1 projektavimo aplinka. Kadangi sistema bus sudaryta iš trijų mazgų, iš kurių du bus identiški, o likęs vienas mazgas turės papildomą USB jungtį, todėl pateikiu tik vieno mazgo principinę schemą. Sistemai bus naudojamas standartinis maitinimo šaltinis.

Principinė schema yra suskirstyta į atskiras dalis, kad būtų aiškiau matoma, kokias funkcijas atlieka kiekviena dalis. Vieno mazgo principinė schema pavaizduota 9 pav. Iš schemos matome, kad ją sudaro tokios dalys:

- mikroprocesorinis valdiklis AT90CAN128. Tai centrinė mazgo dalis atliekanti duomenų apdorojimą;
- a0 dalis, MV prijungimas prie nuoseklios programavimo sąsajos SPI.
- a1 dalis, MV pajungimas prie USB sąsajos. Ši dalis bus realizuota tik viename įrenginyje, tame kuris vykdys stebėjimo duomenų perdavimą į kompiuterį.
- a2 dalis, tai CAN sąsajos prijungimas prie mikroprocesorinio valdiklio per formuotuvą TJA1050. Ši dalis bus kiekviename mazge, nes visi mazgai yra sujungti į bendrą tinklą;
- a3 dalis, tai sensorinė dalis sudaryta iš drėgmės daviklio, kuris matuos drėgmę mazgo patalpinimo vietoje;
- a4 dalis, tai sensorinė dalis sudaryta iš fotodiodo, kuris matuos apšviestumą mazgo patalpinimo vietoje;
- a5 dalis, tai sensorinė dalis sudaryta iš temperatūros daviklio, kuris matuos temperatūrą mazgo patalpinimo vietoje.

Taip pat iš pateiktos schemos (9 pav.) matyti, kad mikroprocesorinio valdiklio galimybės išnaudojamos ne visu pajėgumu, nes lieka laisvi PA0 - PA7, PC0 – PC7, PE0 - PE7 portai bei analoginio skaitmeninio keitiklio PF4 – PF7 įėjimai, kuriuos galima panaudoti tolesniame sistemos vystyme t.y. papildomų sensorių pajungimu ir t.t.



9 pav. Paskirstytos sistemos mazgo principinė schema

5. Paskirstytos matavimo sistemos programavimas

5.1 Programavimo įrankiai

Mikroprocesoriniai valdikliai gali būti programuojami tokiomis programavimo kalbomis: Asembler, C, C++, Java bei kitomis specifinėmis, tam tikrai uždavinių klasei pritaikytomis, programavimo kalbomis (pvz. Ladder‘io diagrama).

Šio darbo MV programavimui bus naudojama C programavimo kalba. Ji pasirinkta dėl to, kad yra universali ir lengvai pritaikoma. Programavimui šia kalba mikroprocesoriniams valdikliams yra sukurta daug programavimo aplinkų (IAR, AVR Studio ir kt.).

Šiam darbui pasirinkta „ATMEL“ firmos sukurta „AVR Studio“ programavimo aplinka. Ši aplinka yra laisvai platinama internete, „ATMEL“ internetinėje svetainėje. „AVR Studio“ naudoja WinAVR įrankių rinkinį. Šiame įrankių rinkinyje aprašytos daugelio mikroprocerinių valdiklių aprašų santraukos. Ši programavimo aplinka pasižymi plačiomis galimybėmis: konkrečiam MV ji leidžia naudoti specifines, su periferija susijusias, funkcijas bei atlikti programos, naudojančios periferiją, veikimo modeliavimą ir realaus laiko derinimą.

CAN sąsajos pranešimų modeliavimui ir derinimui naudojamas „CAN simulator“ įrankis. Šiuo įrankiu galima modeliuoti CAN sąsajos darbą t.y. modeliuoti priimamus ir išsiunčiamus pranešimus, inicializuoti priimtų išsiųstų pranešimų iššaukiamus pertraukimus.

Darbo su CAN sąsaja palengvinimui bus naudojama „ATMEL“ firmos pateikta „CAN library“ biblioteka, kurioje realizuotos pagrindinės CAN sąsajos funkcijos t.y. pranešimų siuntimas, priėmimas ir t.t.

Mikroprocesorinio valdiklio darbui su USB sąsaja užtikrinti bus naudojama „avrusb“ biblioteka, kuri laisvai platinama „Object Development“ internetiniame firmos puslapyje. Ši biblioteka realizuoja programinį 1.1 USB sąsajos standarto palaikymą.

Sistemos mazgų mikroprocesorinių valdiklių užprogramavimui bus naudojama programa „Pony Prog“ palaikanti daugelį rinkoje esančių AVR mikroprocesorinių valdiklių. Šios programos paskirtis – sukompiliuotos programos įrašymas, ištrynimas bei darbo parametrų nustatymas mikroprocesoriniame valdiklyje. „Pony Prog“ programa užprogramuoja MV naudodamasi nuosekliais arba lygiargečiais programatoriais. Šiam darbui bus naudojamas USB programatorius.

Klientinei kompiuterio programai programuoti bus naudojama „Delphi 7“ programavimo aplinka. Ši programavimo aplinka naudoja object pascal programavimo kalbą. Klientinės

programos pagalba bus sekamas sistemos darbas bei užduodamos užduotys paskirstytos sistemos mazgams.

Darbo su USB sąsaja palengvinimui bus naudojama „libusb-win32.dll“ biblioteka. Ši biblioteka yra laisvai platinama.

5.2 CAN sąsajos realizacija AT90CAN128 mikroprocesoriniame valdiklyje

AT90CAN128 mikroprocesoriniame valdiklyje realizuotas CAN sąsajos kontroleris palaikantis 2.0 A ir B specifikacijas. CAN kontrolerio būsena valdoma pasitelkiant registrus:

- CANGCON – pagrindinis 8 bitų CAN kontrolerio valdymo registras. Šio registro pagalba galima CAN kontrolerį nustatyti į tokius rėžimus: įjungtas, išjungtas, pasiklausymo.
- CANGSTA – pagrindinis 8 bitų CAN kontrolerio būsenos registras. Šio registro pagalba galima sužinoti kontrolerio būseną t.y. ar kontrolerio siųstuvas įjungtas, išjungtas ar priimami klaidų pranešimai.
- CANGIE – pagrindinis 8 bitų CAN kontrolerio pertraukimų įjungimo/išjungimo registras; Šio registro pagalba galima įjungti CAN kontrolerio pertraukimus tokiems įvykiams: kontrolerio išjungimui, kontrolerio perėjimui į pasyvią būseną ir kt.
- CANGIT – pagrindinis CAN kontrolerio pertraukimų būsenos registras. Šio registro pagalba galima sužinoti CANGIE registre užduotų pertraukimų būsenas.
- CANTEC – išsiustų pranešimų 8 bitų klaidų skaitiklio registras. Šio registro pagalba kaupiamas išsiustų pranešimų klaidų kiekis. Kai šio registro reikšmė pasiekia 127 CAN kontroleris pervedamas į pasyvią būseną, o kai reikšmė pasiekia 255 kontroleris – deaktyvuojamas.
- CANREC – priimtų pranešimų 8 bitų klaidų skaitiklio registras. Šio registro pagalba kaupiamas priimtų pranešimų klaidų kiekis. Kaip ir su CANTEC registru CAN kontroleris gali būti pervedamas į pasyvią arba neaktyvią būsenas.

CAN sąsajos perduodamų ir priimamų pranešimų valdymas vykdomas per, taip vadinamą, pašto dėžutę, sudarytą iš 15 žinučių objektų (angl. message objects). Žinučių objektai gali būti tokiose būsenose: neaktyvus, perdavimo arba priėmimo. CAN kontrolerio būsena, po MV perkrovimo, yra neapibrėžta. Todėl inicializuojant MV periferinius įrenginius, prieš įjungiant

CAN kontrolerį, šiems žinučių objektams yra būtina tą būseną nustatyti. Jei būsena bus nenustatyta bus neužtikrintas normalus CAN kontrolerio darbas.

CAN žinučių objektų valdymas vykdomas per tokius žinučių objektų registrus:

CANEN2 ir CANEN1 – žinučių objektų įjungimo/išjungimo 8 bitų registrai. Šių registrų pagalba galima įjungti/išjungti reikiamus žinučių objektus.

CANPAGE – aktyvaus žinučių objekto nustatymo 8 bitų registras. Šio registro pagalba nustatomas aktyvus žinučių objektas bei pasikeičia CANIDTn, CANIDMn, CANMSG registrų reikšmės pagal tai, kuris žinučių objektas yra aktyvus.

CANIE2 ir CANIE1 – žinučių objektų pertraukimų 8 bitų registrai. Šių registrų pagalba galima įjungti/išjungti pertraukimus, kuriuos iššauks žinučių objektai.

CANSIT2 ir CANSIT1 – žinučių objektų pertraukimų 8 bitų būsenos registrai. Šių registrų pagalba nustatoma, kuriam iš žinučių objektų suveikė pertraukimas.

CANSTMOB – aktyvaus žinučių objekto būsenos 8 bitų registras. Šio registro pagalba galima sužinoti ar pranešimas priimtas ar išsiųstas bei, kokio ilgio pranešimas priimtas arba išsiųstas.

CANIDT1, CANIDT2, CANIDT3 ir CANIDT4 – aktyvaus žinučių objekto identifikatoriaus 8 bitų nustatymo registrai. Šių registrų pagalba užduodamas priimamo ar išsiunčiamo pranešimo identifikatorius bei nustatoma kaukė kartu su CANIDMn registrais kokie pranešimai bus priimami.

CANIDM1, CANIDM2, CANIDM3 ir CANIDM4 – aktyvaus žinučių objekto identifikatoriaus 8 bitų kaukių (angl. mask) registrai. Šių registrų pagalba galima nustatyti, į kokius CAN pranešimus reaguos žinučių objektas.

CANMSG – aktyvaus žinučių objekto 8 bitų duomenų adreso registras. Šiame registre nurodomas duomenų struktūros adresas, kur patalpinti priimto ar išsiunčiamo pranešimo duomenys.

5.3 Sistemos darbo algoritmas

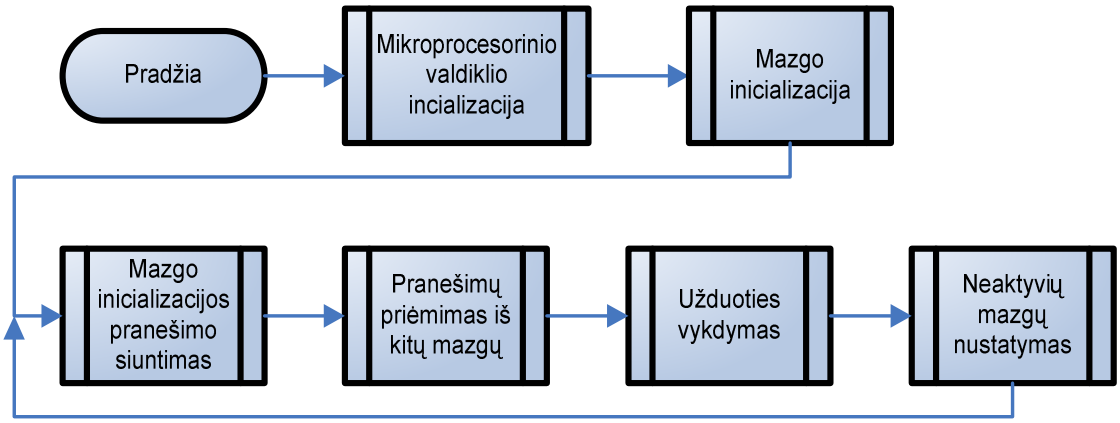
Šioje dalyje bus sudaromi mikroprocesorinių valdiklių sistemos mazgų darbo algoritmai. Mano projektuojama sistema susideda iš trijų analogiškų mazgų, kurie tarpusavyje sujungti CAN sąsaja, tačiau vienas iš mazgų turi papildomą funkcionalumą t.y. ryšį su kompiuteriu per USB sąsają. Todėl sistemos mazgams bus sudaromi du atskiri darbo algoritmai: atskiras darbo algoritmas mazgams, kurie yra be USB ryšio, ir mazgui, kuris turi USB ryšį. Šie algoritmai yra

panašūs, tačiau turi keletą skirtumų, dėl kurių būtina aptarti abu algoritmus. Taip pat svarbu išanalizuoti kaip realizuojama sistemos stebėsena.

Mazgo su USB ryšiu algoritmas ir mazgo be USB ryšio algoritmas aptariami ir vaizduojami kituose poskyriuose.

5.3.1 Periferinio mazgo algoritmas

Mazgo be USB ryšio algoritmas reikalingas tam, kad būtų galima užtikrinti mazgų tarpusavio ryšius (CAN sąsaja) tiek sujungus juos tarpusavyje, tiek po vieną. 10 pav. pateikiu mazgo be USB ryšio algoritmo struktūrinę schemą.



10 pav. Mazgo be USB ryšio algoritmo struktūrinė schema.

Šioje algoritmo struktūrinėje schemoje pavaizduotas nedetalus MV sistemos mazgo algoritmas, todėl kiekvieną iš blokų aptarsiu plačiau.

Mikroprocesorinio valdiklio incializacija. Startavus mikroprocesoriniam valdikliui pirmas etapas, kuris turi būti atliktas t.y. periferijos inicializacija. Šio proceso metu uždraudžiami globalūs pertraukimai bei išjungiama tolimesniame mikroprocesorinio valdiklio darbe nereikalinga periferija bei nustatomi naudojamos periferijos nustatymai.

CAN sąsajos inicializacija. Kadangi CAN sąsaja palaiko kelis duomenų perdavimo greičius, todėl būtina iš pradžių užduoti sąsajoje vykdomų duomenų mainų greitį (nustatytas 250 Kbit/s greitis). Po greičio nustatymo inicializuojami žinučių objektai. Pradžioje visi žinučių objektai nustatomi į neaktyvų rėžimą. Toliau nustatomi naudojami žinučių objektai.

Vienas žinutės objektas nustatomas mazgų inicializacijos pranešimų priėmimams t.y. nustatomas priėmimo rėžimas CMD_RX taip pat suteikiama žinučių priėmimo kaukė (CANIDM1 = 0xF0, CANIDM2 = 0x00, CANIDM3 = 0x00, CANIDM4 = 0x00 ir CANIDT1 =

0x00, CANIDT2 = 0x00, CANIDT3 = 0x00, CANIDT4 = 0x00) t.y. užduodama, kad bus lyginami tik penki jauniausieji identifikatoriaus bitai. Taip pat išvalomas žinutės objekto duomenų buferis.

Vienas žinutės objektas nustatomas mazgo inicializacijos informacijos siuntimui: nustatomas išsiuntimo rėžimas (nustatytas CMD_TX rėžimas), suteikiamas žinutės identifikatorius, kiekvienas mazgas turi unikalų identifikatorių (nuo 0 iki 127). Taip pat į žinutes objekto duomenų buferį patalpinami mazgo inicializacijos duomenys. Mazgo inicializacijos duomenys suformuojami kiekvienam mazgui atskirai t.y. nurodoma, kokį funkcionalumą gali atlikti mazgas bei kiek jis apkrautas.

Vienas žinučių objektas nustatomas stebėjimo duomenų perdavimui į mazgą su USB ryšiu t.y. nustatomas išsiuntimo rėžimas CMD_TX, suteikiamas unikalus identifikatorius (nuo 128 iki 255) ir išvalomas pranešimo duomenų buferis.

Vienas žinutės objektas nustatomas mazgo matavimų užklausoms priimti t.y. nustatomas priėmimo rėžimas CMD_RX, suteikiama žinučių priėmimo kaukė (CANIDM1 = 0xF0, CANIDM2 = 0x00, CANIDM3 = 0x00, CANIDM4 = 0x00 ir CANIDT1 = 0x00, CANIDT2 = 0x20, CANIDT3 = 0x00, CANIDT4 = 0x00). Ši kaukė nustato, kokių identifikatorių pranešimai bus priimami (nustatyta, kad pranešimai bus priimami su identifikatoriais nuo 256 iki 383).

Vienas žinutės objektas nustatomas mazgo funkcionalumo užklausų atsakymams t.y. nustatomas išsiuntimo rėžimas CMD_TX, suteikiamas unikalus identifikatorius nuo 256 iki 383.

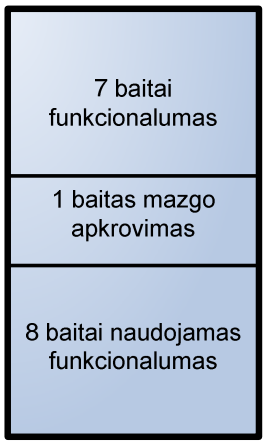
Vienas žinutės objektas nustatomas matavimų užklausoms siųsti t.y. nustatomas išsiuntimo rėžimas CMD_TX, suteikiamas unikalus identitifikatorius pagal mazgo adresą ir išvalomas pranešimų buferis.

Vienas pranešimo objektas nustatomas mazgo atliekamoms užduočių vykdymo užklausoms t.y. nustatomas priėmimo rėžimas CMD_RX , suteikiama žinučių priėmimo kaukė (CANIDM1 = 0xF0, CANIDM2 = 0x00, CANIDM3 = 0x00, CANIDM4 = 0x00 ir CANIDT1 = 0x00, CANIDT2 = 0x30, CANIDT3 = 0x00, CANIDT4 = 0x00). Ši kaukė nustato, kad bus priimami pranešimai su identifikatoriais nuo 384 iki 512.

Inicializavus mikroprocesorinį valdiklį, įjungiamas CAN sąsajos kontroleris. Taip pat įjungiami globalūs pertraukimai.

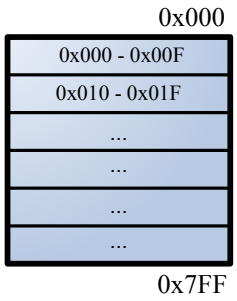
Mazgo inicializacija. Šiame darbe kiekvienas iš sistemos mazgų turės informaciją apie kitus mazgus, todėl kiekvienas mazgas kaups visų mazgų inicializacijos duomenis. Taigi sistema dirbs be jokio vedančiojo įrenginio.

Kiekvieno mazgo inicializacijos duomenims skirta po 8 baitus bei 8 baitai skirti naudojamam nutolusio mazgo funkcionalumui nustatyti. Mazgo inicializacijos duomenys aprašomi tokia struktūra kaip pateikta 11 pav.



11 pav. Mazgo inicializacijos duomenų struktūra.

Numatyta, kad sistema galės palaikyti iki 127 mazgų. Informacija apie kiekvieną mazgą bus laikoma EEPROM (angl. Electrically Erasable Programmable Read-Only Memory) atmintyje. Mazgų inicializacijos duomenims saugoti EEPROM atmintyje yra skirta 2 KB atminties. Šie duomenys talpinami atminties srityje nuo 0x000 iki 0x7FF. Kiekviename mazge inicializacijos duomenys bus saugomi kaip pateikta 12 pav.



12 pav. Mazgų inicializacijos duomenų išdėstymas EEPROM atmintyje.

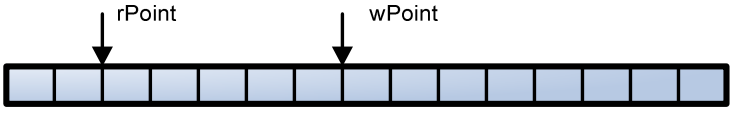
AT90CAN128 mikroprocesoriniame valdiklyje EEPROM atminties kiekis yra 4 KB, atminties adresų erdvė yra nuo 0x0000 iki 0x0FFF. Įrašymas į EEPROM atmintį užima <10 ms vienam baitui. Taip pat turi būti laikomasi tokios nustatytos įrašymo procedūros:

1. Laukiama kol EECR registro bito EEWE reikšmė taps lygi nuliui;
2. Į EEAR registrą įrašomas EEPROM atminties adreso baitas;
3. Į EEDR registrą įrašomas duomenų baitas, kuris bus įrašytas;
4. Įrašomas vienetas į EECR registro EEMWE bitą;
5. Įrašomas vienetas į EECR registro EEWE bitą;

Tarp 4 ir 5 žingsnių turi būti užtikrinta, kad neįvyks pertraukimas, todėl turi būti išjungtas globalus pertraukimų bitas. Iš pateiktos įrašymo procedūros matyti, kad negalima rašyti į EEPROM atmintį vieną po kito baitus nepertraukiamai, nes tai sustabdys mikroprocesorinio valdiklio darbą tokiam n*10 ms (n-baitų kiekis) laikui. Taip pat pabrėžtina, kad EEPROM atminties perrašymų skaičius yra ribotas iki 100000 kartų. Tai reiškia, kad tik tiek kartų galės kisti sistemos konfigūracija.

Kad būtų išvengta MV valdiklio darbo sutrikdymo, neapibrėžtam laikui pasitelkiamas toks metodas: iš pradžių duomenys rašomi į RAM (random access memory) atminties buferį, po to iš šio buferio, po vieną baitą, kiekvieną programos darbo ciklą, įrašoma į EEPROM atmintį taip aklinai nesustabdant MV darbo, bet tik užlaikant jo darbą <10 ms trukmei.

Žemiau pateikiama tokio įrašymo proceso schema (13 pav).



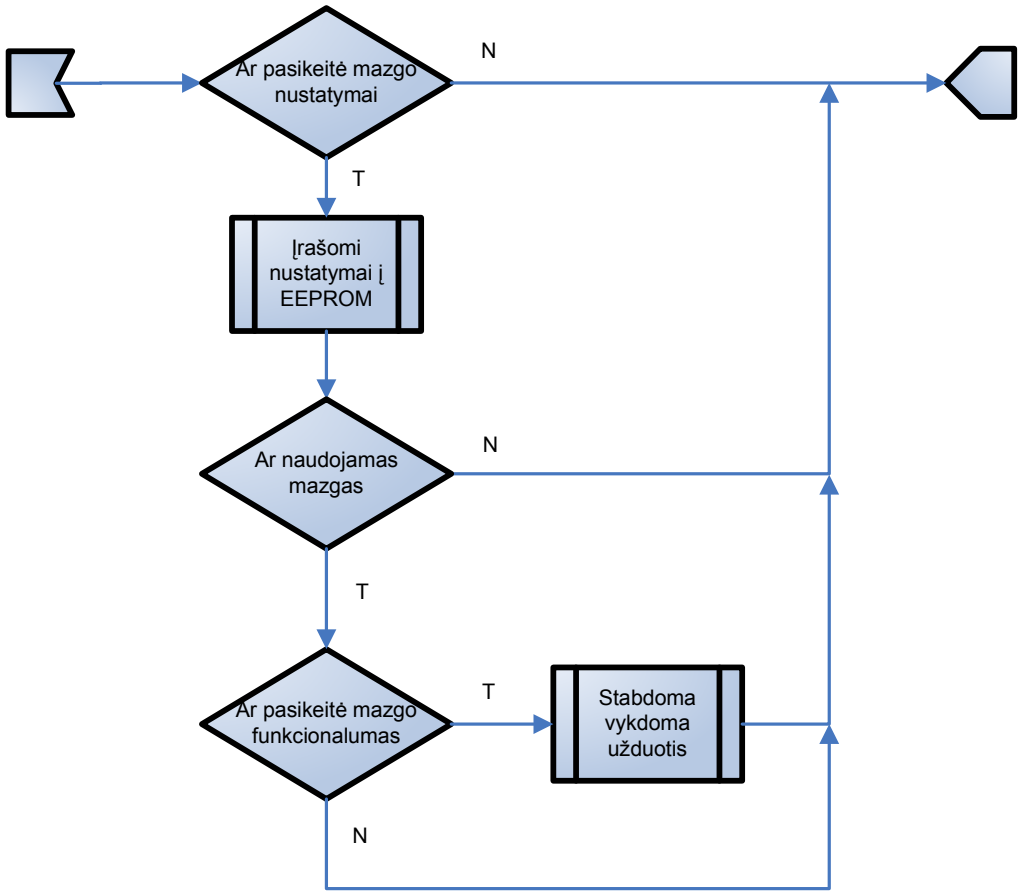
13 pav. Įrašymo į buferį ir skaitymo iš jo schema.

Kaip matyti iš pateiktos schemos, įrašymas į buferį ir įrašymas po baitą į EEPROM atmintį, turi būti suderintas taip, kad wPoint neaplenktų rPoint.

Mazgo inicializacijos pranešimo siuntimas. Kaip jau aprašyta MV inicializacijos etape, jau yra sukonfigūruotas vienas žinutės objektas mazgo inicializacijos duomenų siuntimui. Šiame etape žinutės objekto duomenų buferis užpildomas mazgo inicializacijos duomenimis ir užregistruojamas duomenų siuntimas bei bandomas išsiųsti pranešimas. Nepavykus išsiųsti pranešimo arba gavus išsiuntimo klaidą deaktyvuojamas žinutes objektas bei per naujo inicializuojamas žinutės objektas inicializacijos duomenims išsiųsti. Inicializacijos pranešimai siunčiami kas 100 ms.

Pranešimų priėmimas iš kitų mazgų. Kiekvienas iš sistemos mazgų gali priimti kitų mazgų inicializacijos pranešimus, matavimų užklausų pranešimus ir vykdomų užduočių pranešimus. Aptarsiu atliekamus veiksmus priėmus kiekvieną iš pranešimų tipų atskirai.

Priėmus kito mazgo inicializacijos pranešimą atliekami veiksmai pateikti 14 pav.



14 pav. Struktūrinė algoritmo schema priėmus inicializacijos pranešimą.

Kaip matosi iš pateiktos algoritmo struktūrinės schemos, priėmus mazgo inicializacijos pranešimą patikrinama ar gauti mazgo inicializacijos duomenys sutampa su esančiais EEPROM atmintyje. EEPROM pradžios adresas, nuo kurio reikia nuskaityti 8 baitus, apskaičiuojamas taip – 8*priimto pranešimo identifikatorius.

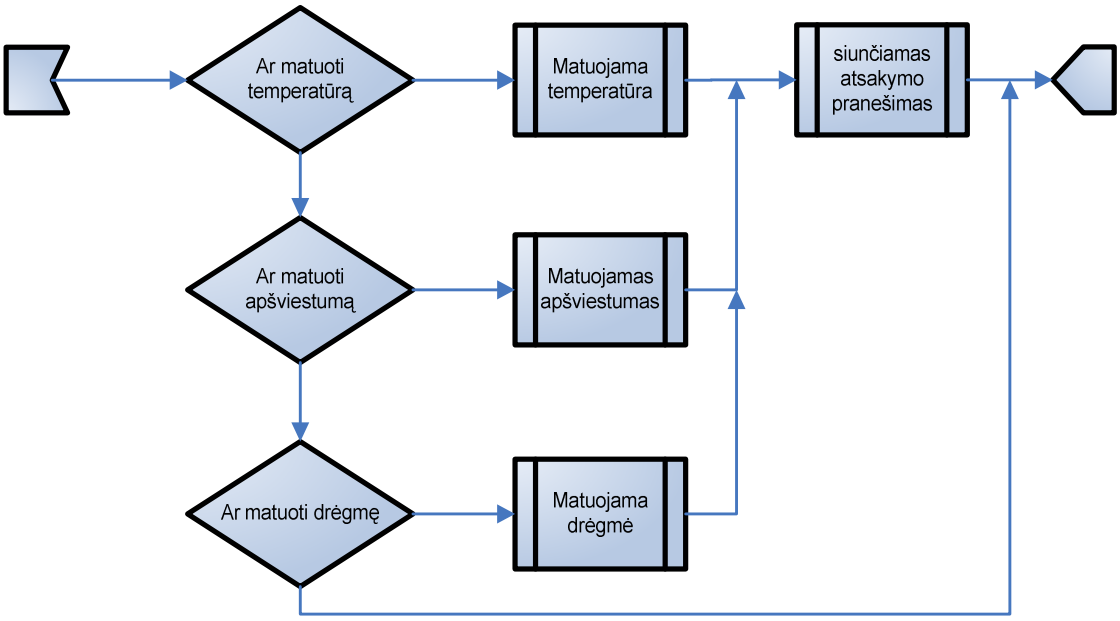
Tam, kad būtų patikrinti mazge esantys kito mazgo inicializacijos duomenys su duomenimis gautais pranešimu, reikia nuskaityti duomenis iš EEPROM atminties.

Kaip jau aprašyta praeituose skyriuose, įrašymo procedūra į EEPROM atmintį užima tam tikrą laiką, todėl ji stabdo MV darbą bei jos metu turi būti užtikrinta, kad neįvyks globalūs pertraukimai. Analogiškai yra ir su skaitymo procedūra, skirtumas tas, kad nuskaitymas yra greitas ir nestabdo MV darbo, tačiau skaitymo metu turi būti užtikrinta, kad neįvyks glabalių pertraukimų. Taip pat nereikia pasitelkti buferio skaitymui po vieną baitą. Skaitymo iš EEPROM atminties procedūra analogiška įrašymo procedūrai, todėl plačiau jos neaptarsiu.

Patikrinus inicializacijos duomenis, jei jie nesutampa, įrašomi pranešimu gauti duomenys į EEPROM atmintį. Toliau patikrinama ar vykdant užduotį naudojamas mazgas, kuriam gautas inicializacijos pranešimas. Jei naudojamas, tikrinama ar nesikeitė naudojamas funkcionalumas,

jei keitėsi, funkcionalumas stabdoma vykdoma užduotis, kurios darbo perskirstymas vykdomas kitame algoritmo cikle.

Priėmus matavimo užklausos pranešimą atliekami veiksmai pateikti 15 pav.



15 pav. Struktūrinė algoritmo schema priėmus matavimo užklausos pranešimą.

Kaip matyti iš pateiktos schemos, priėmus matavimo užklausos pranešimą tikrinama, kokį funkcionalumą užsako pranešimas, pagal tai nustatoma, koks fizikinis dydis turi būti išmatuotas. Temperatūros, apšviestumo ir drėgmės davikliai prijungti prie MV AT90CAN128 analoginio signalo/ skaitmeninį keitiklio (angl. ADC - Analog to digital converter) ASK.

Mano atliekame darbe naudojamos šios jungtys: PF0(ADC0) temperatūrai, PF1(ADC1) apšviestumui ir PF3(ADC3) drėgmėi matuoti.

AT90CAN128 mikroprocesorinis valdiklis turi integruotą ASK. Tai 10 bitų skiriamosios gebos ASK, kuris turi 8 analoginio signalo įvadus (PORTF). ASK gali matuoti įtampos pasikeitimus nuo 0 V iki 5,5 V, priklausomai nuo prijungtos įtampos į AVCC įvadą. ASK yra 10 bitų skiriamosios gebos, todėl ASK duomenų registrai gali priimti 1024 skirtingas įtampos reikšmes. Pvz. Jei matuojama įtampa yra nuo 0 iki 5,5 V tai matuojamos įtampos žingsnis yra 0,005 V.

ASK valdymas vykdomas per registrus. Todėl toliau aprašau pagrindinius registrus.

ADMUX – multiplexerio pasirinkimo 8 bitų registas. Šio registro pagalba nustatomas ASK matuojamos įtampos dydis bei nustatomas, kuris iš PORTF įvadų naudojamas analoginio į skaitmenį signalą konvertavimui.

ADCSRA – ASK kontrolės ir būsenos 8 bitų registras A. Šio registro pagalba įjungiamas/ išjungiamas analoginio signalo keitimas į skaitmeninį.

ADCSRB – ASK kontrolės ir būsenos 8 bitų registras B. Šio registro pagalba nustatoma, koks pertraukimas gali inicializuoti ASK siganalo keitimą.

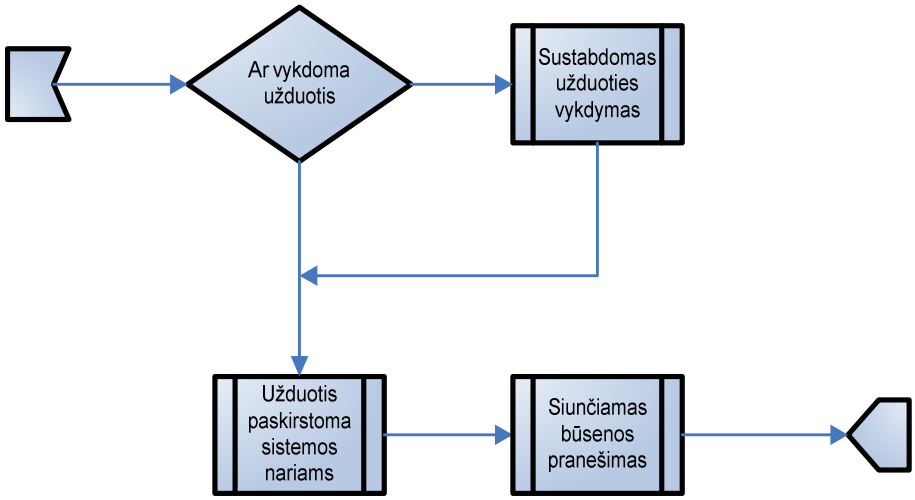
ADCL ir ADCH – ASK duomenų 8 bitų registrai. ADCH registro naudojami tik du vyriausi bitai. Kai ASK keičia analoginį siganalą į skaitmeninį, šiuose registruose skaitmeniniu pavidalu gaunama pakeista signalo reikšmė.

DIDR0 – skaitmeninės įvesties uždraudimo 8 bitų registras. Šio registro pagalba išjungiami skaitmeninės ASK kanalų įvesties buferiai.

Analoginio signalo keitimas į skaitmeninį atliekamas tokia tvarka:

- laukiama kol ADATE arba ADEN bitai registre ADCSRA tampa lygūs nuliui;
- ADMUX registre nustatomas matuojamos įtampos dydis, nustatomas, kuris kanalas naudojamas;
- įrašant vienetą į ADSC registro ADCSRA bitą paleidžiamas analoginio signalo keitimas į skaitmenį;
- laukiama, kol ADIF bito ADCSRA registre reikšmė taps lygi vienetui;
- iš ADCL ir ADCH registrų nuskaitoma pakeista signalo reikšmė.

Priėmus vykdomų užduočių pranešimą atliekami veiksmai pateikti 16 pav.

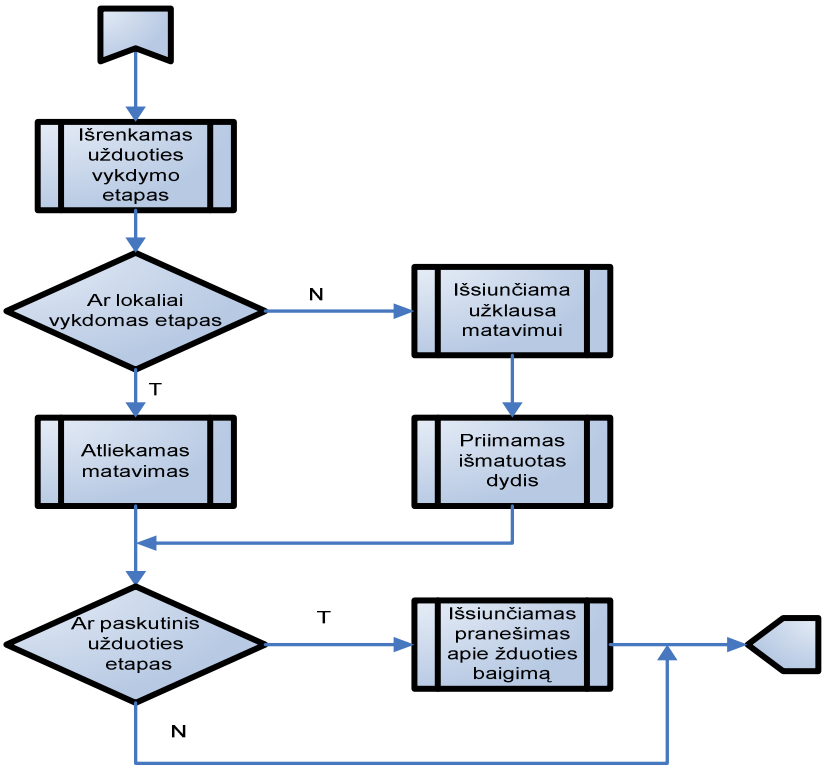


16 pav. Struktūrinė algoritmo schema priėmus vykdomų užduočių pranešimą.

Kaip matyti iš pateiktos schemos, pradžioje patikrinama ar MV vykdo, kokia nors užduotį. Jei vykdo – sustabdomas užduoties vykdymas, jei ne – pereinama prie užduoties paskirstymo sistemos nariams.

Užduoties skirstymas sistemos nariams. Kadangi kiekvienas iš mazgų gali atlikti jam patikėtą užduoti bei pasitelkti kitus MV sistemos narius, todėl būtinas užduoties paskirstymo algoritmas. Gavus užduotį MV pasitikrina ar pats gali atlikti užduotį, jei gali – nusistato užduoties atlikimui, jei ne – bando išskirstyti užduotį visiems sistemos mazgams tokiu principu: randa mažiausiai apkrautą sistemos mazgą bei tikrina jo funkcionalumą, jei šis mazgas turi užduoties vykdymui tinkamo funkcionalumo, tai priskiria vieną funkciją, kurią turės atlikti mazgas. Toliau einama prie sekančio sistemos mazgo ir dalinama užduotis tol kol visiškai išdalinama. Nepavykus išdalinti užduoties ar išdalinus sėkmingai užduotį, siunčiamas būsenos pranešimas mazgui su USB sąsaja.

Užduoties vykdymas. Paskirsčius gautą užduotį pradedamas užduoties vykdymas. Užduoties paskirstymas, kuris patalpintas prie kiekvieno mazgo inicializacijos nuskaitomas ir pasidedamas į laikinąjį buferį RAM atmintyje, tam kad neskaitytų kiekvieną kartą iš EEPROM atminties, nes tai stabdo MV darbą. Po to kiekvienu programos darbo ciklu susirandamas pagal eiliškumą reikalingas vykdyti užduoties etapas ir bandomas įvykdyti. Įvykdžius visus užduoties etapus siunčiamas pranešimas į mazgą su USB ryšiu. Užduoties vykdymo algoritmo struktūrinė schema pavaizduota 17 pav.



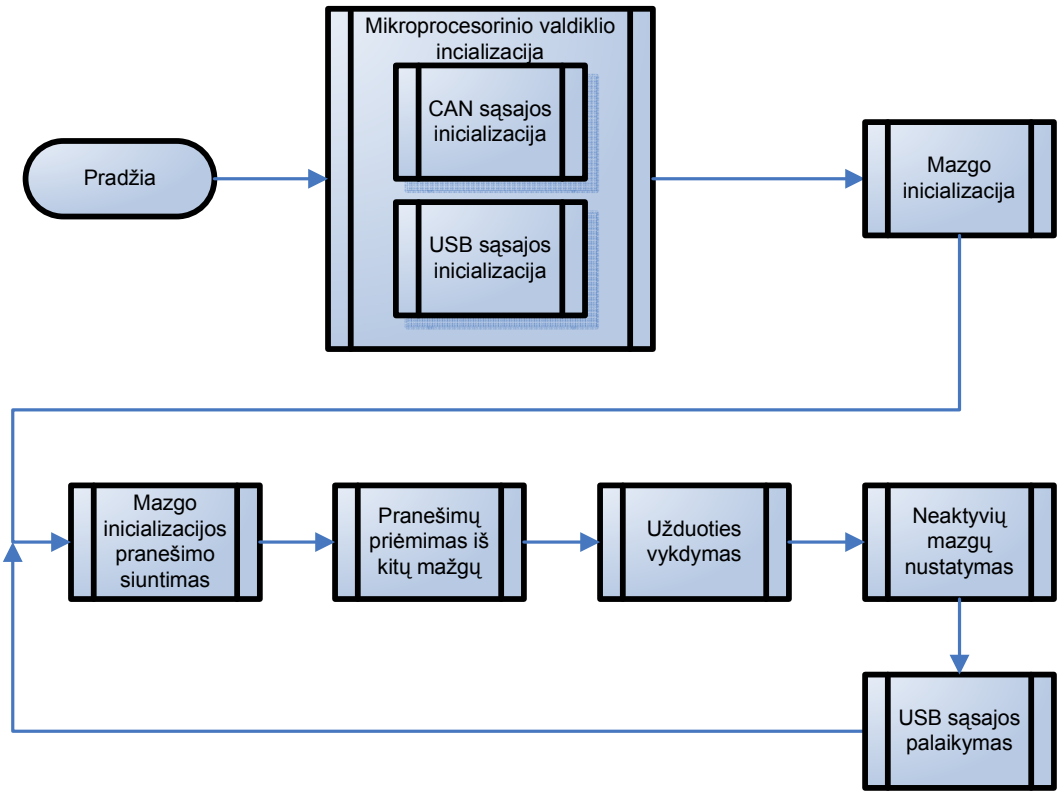
17 pav. Užduoties vykdymo algoritmo struktūrinė schema

Neaktyvių mazgų nustatymas. Kiekvienas sistemos mazgas turi informaciją apie visus kitus sistemos mazgus, todėl gali sekti, kurie iš mazgų yra aktyvūs, o kurie sugedę.

Kiekvienas mazgas savo inicializacijos duomenis siunčia kas 100 ms. Mazgas nustato ar kitas mazgas aktyvus tokiu būdu: gavęs pirmą kart kito mazgo inicializacijos duomenis nustato jam skaitliuką, kuris, kol negaunami nauji inicializacijos duomenys iš mazgo, kas kiekvieną programos darbo ciklą mažinamas ir kai pasiekia 0 skaitoma, kad mazgas atsijungė, o kai gaunami mazgo inicializacijos duomenys skaitiklis nustatomas į 1023. Kai nustatoma, kad bent vienas mazgas atsijungė, patikrinama ar nenaudojamas tas mazgas matavimams, jei naudojamas – sustabdomas užduoties vykdymas ir išsiunčiamas klaidos pranešimas į mazgą su USB sąsaja. Po to inicializuojamas užduoties vykdymo perskirstymas.

5.3.2 Centrinio mazgo algoritmas

Mazgo su USB ryšiu algoritmas reikalingas tam, kad būtų galima užtikrinti mazgų tarpusavio ryšius (CAN sąsaja) tiek sujungus juos tarpusavyje, tiek po vieną. Taip pat būtų galima stebėti bendrą mikroprocesorinių valdiklių sistemos darbą, perduodant sistemos būseną, per USB sąsają. 18 pav. pateikiu mazgo su USB ryšiu algoritmo struktūrinę schemą.



18 pav. Mazgo su USB ryšiu algoritmo struktūrinė schema.

Kaip matyti iš XXpav., mazgo su USB sąsaja algoritmas yra panašus į mazgo be USB sąsajos, todėl struktūrinių dalių kuriose algoritmas yra toks pats neaptarsiu, aptarsiu tik tas dalis kurios yra skirtingos.

Mirkroprocesorinio valdiklio inicializacija. Kaip ir kituose mazguose, taip ir šiame reikia inicializuoti CAN sąsają. CAN sąsajos inicializacija yra beveik identiška, tačiau sukonfiguruojamas papildomas žinutės objektas informacijos perdavimui sistemos stebėsenai.

Žinutės objektas sistemos stebėsenai t.y. nustatomas priėmimo rėžimas CMD_RX, suteikiama žinučių priėmimo (CANIDM1 = 0xF0, CANIDM2 = 0x00, CANIDM3 = 0x00, CANIDM4 = 0x00 ir CANIDT1 = 0x00, CANIDT2 = 0x10, CANIDT3 = 0x00, CANIDT4 = 0x00). Ši kaukė nustato, kokių identifikatorių pranešimai bus priimami (nustatyta, kad bus priimami pranešimai su identifikatoriais nuo 128 iki 255).

USB sąsajos inicializacija. Kadangi naudojama tik programinis USB sąsajos palaikymas, todėl ir inicializacija yra sudėtingesnė. Nustatoma, prie kokio prievado (PORTD) pajungtos USB sąsajos duomenų linijos (nustatyta D- linija prie PD1, o D+ linija prie PD0). Nustatoma, kokiu greičiu dirbs MV (nustatyta 16 Mhz). Toliau įrenginiui suteikiamas unikalus identifikatorius. Nustatoma, kokios funkcijos bus naudojamos. Bus naudojamos usbFunctionRead ir usbFunctionWrite funkcijos. usbFunctionRead funkcija – naudojama duomenų perdavimui iš MV į kompiuterį, o usbFunctionWrite funkcija – naudojama duomenims perduoti iš kompiuterio į MV. Toliau nustatomas įrenginio pavadinimas (pasirinktas „CAN Monitor“ pavadinimas). Toliau aprašomas hidDescriptor, kuriame nurodomas, kokio dydžio duomenų blokais bus perduodami duomenys (nustatyta 128 baitų blokais), nustatoma kiek ir kokie pranešimų tipai bus naudojami (nustatyta, kad bus naudojami dviejų tipų pranešimai USBRQ_HID_GET_REPORT ir USBRQ_HID_SET_REPORT).

Pranešimų priėmimas iš kitų mazgų. Kaip ir kituose mazguose šiame mazge apdorojami visi anksčiau aprašyti pranešimai, bet dar papildomai apdorojami pranešimai, kurie skirti informacijos perdavimui į USB. Gavus tokio tipo pranešimą, informacija, esanti jame, rašoma į buferį. Informacija yra lygiagrečiai paimama per USB sąsają. Duomenų paėmimą ir perdavimą USB sąsaja aprašysiu vėliau.

USB sąsajos palaikymas. Norint užtikrinti stabilų USB sąsajos darbą ne rečiau kaip 50 ms pranešimai turi būti siunčiami į USB sąsaja.

Duomenų perdavimas USB sąsaja. USB sąsajos veikimas pagrįstas pertraukimo inicializavimu ir vienos iš funkcijos (usbFunctionRead arba usbFunctionWrite) vykdymu.

Duomenų nuskaitymas iš įrenginio vykdomas lygiagrečiai pagrindinei programai, todėl, norint užtikrinti duomenų mainus, naudojamas papildomas bufferis. Kaip ir su EEPROM

atmintimi, taip ir su USB buferiu, skaitymas ir įrašymas į buferį turi būti suderintas taip, kad neaplenktų vienas kito.

Kai kompiuteryje esanti programa pareikalauja duomenų, tuomet vykdoma usbFunctionRead funkcija, o kai perduodami duomenys į MV tada vykdoma usbFunctionWrite funkcija.

5.4 Sistemos konfigūracija

Sistema susideda iš trijų mazgų, kiekvienas sistemos mazgas siunčia inicializacijos duomenis kitiems mazgams, pagal tai, kiekvienas mazgas žinos apie visus kitus mazgus.

Kiekvienam mazgui suteikiamas unikalus identifikatorius:

- Centriniam mazgui – ID = 0;
- 1 periferiniam mazgui – ID = 1;
- 2 periferiniam mazgui – ID = 2.

Visi sistemos mazgai, išskyrus centrinį mazgą, stebėjimo duomenų perdavimui naudoja žinučių objektą. Šio žinutės objekto pagalba stebėjimo duomenys siunčiami į centrinį mazgą.

Šiems žinučių objektams suteikti tokie identifikatoriai:

- 1 periferiniam mazgui – ID = 129;
- 2 periferiniam mazgui – ID = 130.

Visi sistemos mazgai matavimo duomenų perdavimui į kitus mazgus naudoja žinučių objektą. Kiekvienam žinučių objektui suteikiamas unikalus identifikatorius.

Žinučių objektams buvo suteikti tokie identifikatoriai:

- Centriniam mazgui – ID = 256;
- 1 periferiniam mazgui – ID = 257;
- 2 periferiniam mazgui – ID = 258.

Visi sistemos mazgai, išskyrus centrinį mazgą, vykdomų užduočių perdavimui naudoja žinučių objektą. Šio žinučių objekto dėka kiekvienas sitemos mazgas gaus užduotis.

Šiems žinučių objektams suteikti šie identifikatoriai:

- 1 periferiniam mazgui – ID = 385;
- 2 periferiniam mazgui – ID = 386.

5.5 Klientinė programa

Paskirstytos mikroprocesorinių valdiklių sistemos stebėjimui bei užduočių perdavimui naudojama klientinė programa.

Klientinė programa parašyta Delphi programavimo aplinka. Ryšio su USB sąsaja palengvinimui naudojama „libusb-win32.dll“ biblioteka. Šios bibliotekos pagalba paprasta rasti prie USB sąsajos prijungtus įrenginius, prisijungti bei inicializuoti duomenų perdavimą tiek iš kompiuterio į įrenginį, kuris yra prijungtas prie USB sąsajos, tiek atvirkščiai.

Klientinės programos pagalba galima atlikti tokius veiksmus:

- Stebėti, prie paskirstytos sistemos prisijungusius, įrenginius;
- matyti įrenginių teikiamas funkcijas;
- suformuoti bei perduoti užduotis sistemos mazgams;
- stebėti kiekvieno iš mazgų atliekamas užduotis (šiuo atveju – matavimus);
- stebėti sistemoje įvykusias klaidas;
- stebėti, tarp sistemos mazgų, perduodamus pranešimus.

Klientinės programos algoritmas. Prisijungus prie USB įrenginio sukuriama gija, kurios pagalba kas 20 ms siunčiama užklausa duomenims iš įrenginio gauti. Gavus duomenis iš įrenginio, informuojami visi objektai, kurie yra prisiregistravę duomenų gavimui. Kiekvienas iš objektų gavęs pranešimą apdoroja tik jam skirtus pranešimus. Kiekvienam mazgui suformavus užduotį perduodamas užduoties pranešimas į USB įrenginį.

5.6 Iškilusios problemos

Atliekant šį darbą problemų iškilo tokiose srityse:

1. mikroprocesorinio valdiklio darbo parametrų nustatyme;
2. įrašyme bei skaityme iš EEPROM atminties;
3. CAN pranešimų konfigūracijoje.

Kiekvieną iš probleminių sričių aptarsiu detaliau bei nurodysiu problemų sprendimo būdus.

Mikroprocesorinio valdiklio darbo parametrų nustatymas. Kiekvienas MV turi savo darbo parametrus, tokius kaip: taktinio dažnio generatoriaus pasirinkimas bei nustatymas (AT90CAN128 MV tai CKSEL3...CKSEL0 bitai), dažnio daliklis (CKDIV8).

MV gali dirbti su kelių tipų rezonatoriais: išoriniais (kvarcinis rezonatorius, RC generatorius arba išorinis generatorius), vidinis rc generatorius (AT90CAN128 MV jis dirba

8Mhz dažniu). Programų ir duomenų atminties užrakinimo bitai (AT90CAN128 MV tai BLB12...BLB01 ir LB2,LB1 bitai). Bei kiti konkretaus MV darbo parametrai.

Užprogramuojant MV darbo parametrus buvo klaidingai nustatyti taktinio dažnio rezonatoriaus parametrai, ko pasekoje, MV neleisdavo įrašyti, trinti ar kitaip pasiekti programų atminties. Iškilusios problemos sprendimas buvo rastas mikroprocesorinių valdiklių programuotojų forumuose (www.avrfreaks.net, www.nabble.com, www.allaboutcircuits.com) t.y. užprogramuojant parametrus dažniausiai daroma klaida – sumaišomos bitų reikšmės, nes AVR MV parametrų bituose „0“ reikšmė laikoma užprogramuota, „1“ – neužprogramuota, todėl, kai blogai užprogramuojami taktinio dažnio generatoriaus bitai, dažniausiai jie būna nustatomi ant išorinio generatoriaus. Problema išspręsta pasigaminus 1Khz išorinį generatorių ir prijungus jį prie MV generatoriaus įėjimų bei nustačius rezonatoriaus parametrus taip, kad būtų naudojamas vidinis RC generatorius. Paskui prijungus kvarcinį rezonatorių teisingai nustatyti bitai.

EEPROM atminties įrašymas bei skaitymas.. Sistemos mazgų konfigūracijos laikomos EEPROM atmintyje, todėl neišvengiamai reikia įrašinėti daugiau nei po vieną baitą į EEPROM atmintį. Dažnai veikimo metu sistema neprognozuojamai išeidavo iš rikiuotės, todėl to priežasties buvo ieškoma, išjungiant kodo fragmentus. Priežastis buvo rasta: kai rašoma į EEPROM atmintį sistema persikrauna. Nagrinėjant „ATMEL“ pateiktos darbui su EEPROM bibliotekos procedūras bei AT90CAN128 mikroprocesorinio valdiklio specifikacijas nustatyta, kad įrašymo procedūrose neužtikrinamas globalių pertraukimų uždraudimas. Pakeitus procedūras problema išsisprendė iš dalies. Įrašant į EEPROM stabdomas MV darbas, o to pasekoje sutrikdavo USB sąsajos palaikymas, todėl pasitelktas buferis ir įrašymo procedūra aprašyta periferinio mazgo darbo algoritmo skyriuje.

CAN pranešimų konfigūracija. Kiekvienas pranešimas konfigūruojamas kaip žinutės objektas kontroleryje. Žinutės objektams galima nustatyti priimamų pranešimų kaukes, kurios nusistato CANIDM1...CANIDM4 ir CANIDT1...CANIDT4 registruose. Kadangi MV CAN sąsajos kontroleris palaiko 2.0 A (11 bitų identifikatorius) ir B (29 bitų identifikatorius) sąsajos standartus, o CANIDMn ir CANIDtn registruose skirtingai išdėstomi kaukės bitai kiekvienam iš šių standartų, todėl tampa komplikuotas kaukės uždėjimas žinutės objektui. Sprendimas rastas forumuose bei atidžiai perskaičius AT90CAN128 MV specifikaciją. Pasirodo, naudojant B standarto kaukes, reikia nepamiršti CANCDMOB registro bite IDE nustatyti, kokio standarto identifikatoriai naudojami.

Rezultatai ir išvados

Suprojektuota ir pagaminta paskirstytų mikroprocesorinių valdiklių sistema:

- Parinkta sistemos struktūra ir naudojama sąsaja (CAN sąsaja).
- Parinktas sąsajos protokolas (CANKingdom).
- Pasirinktas mazgo mikroprocesorinis valdiklis (AT90CAN128).
- Suprojektuota principinė elektrinė mazgo schema, pagal ją atliktas trasavimas ir pagaminti 3 mazgai.
- Parašytos mazgų darbo programos.
- Sistemos veikimo stebėjimui viename mazge panaudota USB sąsaja, per kurią į kompiuterį perduodami sistemos konfigūracijos pakitimai ir darbo sutrikimai. Kompiuteriui parašyta klientinė programa, sekanti sistemos darbą.
- Atlikti eksperimentai ir sistemos algoritmas pakoreguotas taip, kad sistemos darbas nesutriktų, atsijungus vienam iš mazgų, bei atsijungusio mazgo funkcionalumą perimtų mazgai turintys perteklinį funkcijų kiekį.

Atlikus šį darbą galima padaryti tokias išvadas:

- Perteklinių funkcijų panaudojimas, paskirstytų mikroprocesorinių valdiklių sistemų mazguose, padėjo padidinti tokios sistemos atsparumą trikdžiams (mazgų gedimams, mazgų sudedamųjų dalių gedimams) bei užtikrinti stabilų sistemos darbą.
- Paskirstytos MV sistemos be vedančiųjų mazgų pranašesnės už sistemas su vedančiuoju mazgu, tačiau tokioje sistemoje yra daug sudėtingiau užprogramuoti sistemos mazgus.

Pasiūlymas:

Norint patobulinti esamą sistemą, būtų galima pasitelkti bevielo ryšio technologijas bei išanalizuoti tokių ryšių pranašumus ir trūkumus. Tokios technologijos įvedimas suteiktų patogesnę, sistemos mazgų vietos ir atstumo tarp jų pakeitimo, galimybę.

Šaltinių sąrašas

[Art02] HASSAN A. ARTAIL. A distributed system of network-enabled microcontrollers for controlling and monitoring home devices. 2002

[Atm07] Atmel, AT90CAN32/64/128 datasheet, 2007

[Bar03] Michael Barr. "Choosing an RTOS", Embedded Systems Programming, January 2003.

[CBB05+] J.O. Coronel, F. Blanes, G. Benet, J.E. Simo, P. Perez, M. Albero. CAN-based Distributed Control Architecture Using the SCoCAN Communication Protocol. Departamento de Informatica de Sistemas y Computadores, 2005

[Das08] Ranjan Dasgupta. Anatomy of RTOS and Analyze the Best-Fit for Small, Medium and LargeFootprint Embedded Devices in Wireless Sensor Network. 2008

[Dek00] V. Deksnys. Skaitmeninės sąsajos. KTU Elektroninių ir matavimo sistemų katedra. 2000 [žiūrėta 2007-03-25]. Prieiga per internetą <http://www.msl.ktu.lt/courses/KompKomunikacijos/KompKomunikacijuPaskaitos.pdf >.

[DJ00] Vytautas Deksnys, Vaclovas Jastramskas. Įterptinės sistemos. Kauno technologijos universitetas, Kaunas Technologija, 2000 [žiūrėta 2007-10-14] Prieiga per internetą http://www.msl.ktu.lt/courses/1dalis.pdf

[ED02] EUROTHERM DRIVES. Profibus-DP Communications Interface Technical Manual. 2002

[Gof00] Donald Goff. Choosing the right fieldbus. National Instruments, 2000

[Hri07] Anton Hristozov. Anatomy of an RTOS for small devices. 2007

[Ibr06] Dogan Ibrahim. Microcontroller Based Applied Digital Control. Department of

Computer Engineering Near East University, Cyprus, 2006

[JTN05] Karl Henrik Johansson, Martin Torngren, Lars Nielsen. Vehicle Applications of Controller Area Network, 2005

[MM96] MODICON. MODBUS Protocol Reference Guide Rev J. June 1996

[Phi07] Philips. I2C-bus specification and user manual Rev. 03 — 19. June 2007

[Rey03] Stéphane Rey. Introduction to LIN (Local Interconnect Network) Revision 1.0. May 13th, 2003

[RL03] John Rinaldi, Vince Leslie. The Fast Guide to Controller Area Network (CAN) Application Layers, 2003

[SB02] AV Scott and WJ Buchanan Napier University, Edinburgh, UK. Truly Distributed Control Systems using Fieldbus Technology, 2002

[Sie96] Siemens. Controller Area Network. 1996

[Ti04] Texas Instruments. Comparing Bus Solutions, February 2004

[Tiar02] Texas Instruments Application Report. Introduction to the Controller Area Network (CAN). August 2002

Santrumpos

ASK	Analoginio signalo į skaitmeninį keitiklis (angl. Analog to digital converter adc)
AVR	Atmel firmos 8 bitų mikroprocesorinių valdiklių šeima
CISC	Complex Instruction Set Computer (kompiuteris su pilnu komandų rinkiniu)
COB	Communication object (komunikacijos objektas)
CRC	Cyclic Redundancy Checksum (perteklinės ciklinės sumos liekana)
DMA	Direct Memory Access (tiesioginė prieiga prie atminties)
EDS	Electronic Data Sheet (elektroninė duomenų lentelė)
EEPROM	Electrically Erasable Programmable Read-Only Memory (elektriškai ištrinama programuojama pastovios programos atmintis)
MV	Mikroprocesorinis valdiklis
OSI	Open Systems Interconnection Model (abstraktus ryšio protokolų, naudojamų ryšio ir kompiuteriniuose tinkluose aprašymas)
PDO	Process Data Object (valdymo duomenų objektas)
QFN	Quad Flat Package (kvadratinis plokščias korpusas)
RAM	Random-access memory (operatyvioji atmintinė)
RISC	Reduced Instruction Set Computer (kompiuteris su sumažintu komandų rinkiniu)
RTOS	Real-Time Operating System (realaus laiko operacinė sistema)
SDO	Service Data Object (duomenų serviso objektas)
SOF	Start of frame bit (kadro pradžios bitas)
STP	Shielded twisted pair (apsaugota vyta pora)
TCP/IP	Transmission control protocol /Internet Protocol (perdavimo valdymo protokolas / interneto protokolas)
USB	Universal Serial Bus (universali nuoseklioji jungtis)
UTP	Unshielded twisted pair (neapsaugota vyta pora)
VDD	Positive supply voltage (teigiama maitinimo įtampa)
SPI	Serial programming interface (nuoseklioji programavimo sąsaja)
MSP430	Texas instruments firmos 16 bitų mikroprocesorinių valdiklių šeima
ARM7	ARM Holdings firmos 32 bitų mikroprocesorinių valdiklių šeima
H8	Hitachi firmos 8 ir 16 bitų mikroprocesorinių valdiklių šeima