**TTÜ Robotiklubi MTÜ**

12

**ITKiller kommunikatsioon**

**Kommunikatsiooniprotokoll**

**TTÜ Robotiklubi MTÜ**

**Sisukord**

[Arhitektuur 3](#_Toc347650846)

[CAN liides 4](#_Toc347650847)

[Jadaliidese 4](#_Toc347650848)

[Sõnumid ja signaalid 4](#_Toc347650849)

[Sõnumid 5](#_Toc347650850)

[0xD0 - MotorSpeeds 6](#_Toc347650851)

[0xD1 – Motor1Status, 0xD2 – Motor2Status ja 0xD3 – Motor3Status 6](#_Toc347650852)

[0xD4 - MotorPID 7](#_Toc347650853)

[0xB0 – BallControl 7](#_Toc347650854)

[0xB1 – BallStatus 7](#_Toc347650855)

[0xB2 – LineStatus 8](#_Toc347650856)

[0xC0 – UIControls 8](#_Toc347650857)

[0xC1 – UIFeedback 9](#_Toc347650858)

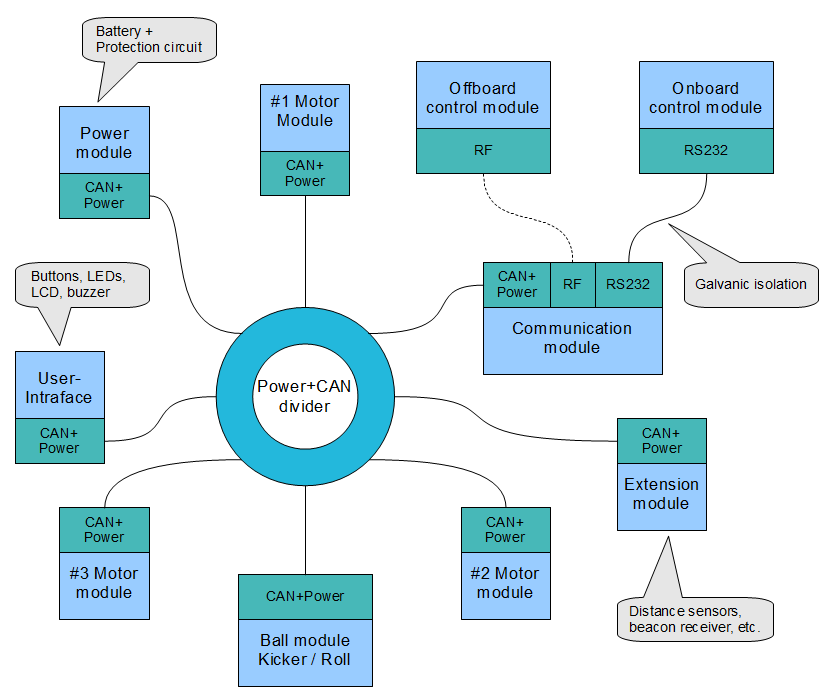
[0xC2 – UIScreenClear 9](#_Toc347650859)

[0xC3 – UIScreenWrite 10](#_Toc347650860)

[0xA0 - PowerStatus 10](#_Toc347650861)

[0xF0 – Sensors 10](#_Toc347650862)

# Arhitektuur



Roboti “targad” moodulid suhtlevad omavahel CAN siinil. Et PC saaks CAN-iga ühendada, on CAN siinil spetsiaalne kommunikatsioonimoodul mis teisendab CAN sõnumid UART sõnumiteks ja vastupidi:

UART siinil on kommunikatsiooniprotokolli aluseks võetud CAN. Ehk teisisõnu UART siinil edastatakse peaaegu sama struktuuri ja sisuga sõnumeid nagu CAN siinil. Vahe on ainult sõnumi paketi ülesehituses, mis CAN-i puhul on standardiga ära määratud, UART-i puhul tehtud selles dokumendis.

## CAN liides

Kasutusel on CAN 2.0A standard. Seega ID väli on 11 bitine, kuid kasutatal on sellest ainult 8 madalamat bitti. Kolm kõrgemat bitti on 0. Bitikiirus on 250 kbit/s. Sampling point 75%.

## Jadaliidese

UART/RS232 siinil edastatakse mõlemas suunas samasuguse ehitusega pakette. Pakett mis on saadetud PC-st jadaliidese kaudu CAN siinile, sealt tagasi ei saadeta (ei peegeldu).

|  |  |  |
| --- | --- | --- |
| **Bait** | **Sisu** | **Väärtus** |
| 0 | Preambula | 0xAA |
| 1 | Sõnumi ID | 0x00 - 0xFF |
| 3 | Sisu pikkus baitides (N) | 1-8 |
| 4 | Sisu, ehk signaalid (1..N baiti) | N \* 0x00 – 0xFF |
| 5 |
| … |
| 3+N |
| 4+N | Kontrollsumma (16-bit CRC) | 0x00 – 0xFFFF |
| 5+N |

Preambula (väärtus 0xAA) on kasutusel sõnumi alguse leidmiseks jadaliidese andmevoost. Preambulale järgneb sõnumi identifikaator (ID) mis on üks-ühele vastavuses CAN sõnumite ID väljaga (8 madalamat bitti sellest). Sõnumi sisu pikkus on 1 bait, millel võib olla väärtuseks 1-8, muul juhul loetakse sõnum vigaseks. Sõnumi sisu on määratud sõnumi ID-ga. Sisus peab olema niipalju baite kui pikkuses kirjas on.

Paketi lõpus on kogu kontrollsumma, millesse arvutatakse sisse sõnumi ID, sisu pikkus ja sisu baidid. CRC algoritm on veel defineerimisel. Arvutatud kontrollsumma ja paketis sisalduvad kontrollsumma mittekattumisel sõnumit ignoreeritakse.

## Sõnumid ja signaalid

Sõnumitel puudub saatja ja addressaat, selle asemel on sõnumil identifikaator (ID) mille alusel sõnumit ja selle sisu tuvastada. CAN ja UART siinil jõuavad kõik sõnumid kõigi mooduliteni. Iga moodul otsustab kas ta on sõnumist huvitatud või mitte. Üldjuhul on siiski määratud mis moodul mis sõnumit saadab.

Sõnumi sisuks on signaalid, ehk ainuühikulised andmed. Signaal võib olla näiteks nupu olek, mootori kiirus või midagi muud mis tähistab ühe kindla funktsiooni reaalset või soovitud olekut. Seega saab sõnumitega edastada olekusignaale ja juhtsignaale.

Signaalide andmehulk on bitipõhine ja nende jaoks on sõnumis reserveeritud täpselt vajalik hulk bitte. Sõnumite kirjelduses on signaalide juures märgitud nende algusbitt ja pikkus. Baidis loendatakse bitte alates vanimast. Ehk näiteks 2-bitine signaal, mille algbiti number on 10 asub teise baidi kolmandas ja neljandas bitis. Sõnumist signaali leidmiseks oleks selle mask 0xC00.

Kokku on ühes sõnumis 1-8 baiti, ehk 8-64 bitti. Bitte loendatakse alates nullist.

# Sõnumid

Järgnevalt on loetletud kõik roboti moodulitevahelised sõnumid

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Sõnumi nimi | Selgitus | Saatja | Periood |
| 0xD0 | MotorSpeeds | Kolme mootori sõidukiiruse ühissõnum. | PC | 10 ms |
| 0xD1 | Motor1Status | Esimese mootori hetkekiirus, vool, temperatuur. | Mootor 1 | 100 ms |
| 0xD2 | Motor2Status | Teise mootori hetkekiirus, vool, temperatuur. | Mootor 2 | 100 ms |
| 0xD3 | Motor3Status | Kolmanda mootori hetkekiirus, vool, temperatuur. | Mootor 3 | 100 ms |
| 0xD4 | MotorPID | PID konstantide P, I ja Dmääramine | PC | - |
| 0xB0 | BallControl | Pallimooduli juhtsignaalid. | PC | 20 ms |
| 0xB1 | BallStatus | Pallimooduli olekusignaalid. | Pallimoodul | 20 ms |
| 0xB2 | LineStatus | Eesmiste jooneandurite olek | Pallimoodul | 20 ms |
| 0xC0 | UIControls | Nuppude olekusignaalid. | Komm. moodul | 100 ms |
| 0xC1 | UIFeedback | Indikaator LED-ide ja kõlari juhtsignaal. | PC | - |
| 0xC2 | UIScreenClear | Ekraani kustutamine. | PC | - |
| 0xC3 | UIScreenWrite | Ekraanile kirjutamine. | PC | - |
| 0xA0 | PowerStatus | Aku seisukord. | Komm. moodul | 1000 ms |
| 0xF0 | Sensors | Sensorite tulemused |  | 20 ms |

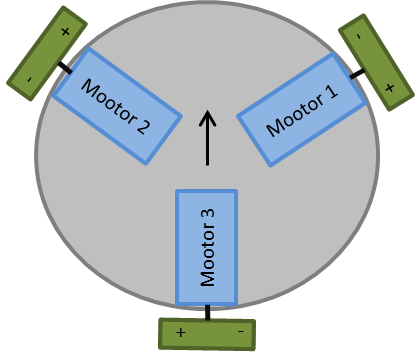
Iga sõnum on eraldi lahti seletatud. Sõnumi periood näitab aega, mille tagant sõnumit peab (peaks) kindlaksmääratud saatja seda saatma. Perioodid tuleb ka reaalse sideliidese läbilaskevõimega vastavusse seada. Kui moodulid, mis ootavad kindlat sõnumit, ei saa seda 10 kordse nominaalperioodi jooksul võivad nad ohutuse mõttes välja lülituda, seisma jääda vmt. Sõnumeid, millel pole perioodi määratud, saadetakse vastavalt vajadusele.

## 0xD0 - MotorSpeeds

Mootorite kiiruste seadmise sõnum.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 16 | int16 | rpm | Motor1RequestSpeed | Mootori 1 soovitud pöörlemiskiirus |
| 16 | 16 | int16 | rpm | Motor2RequestSpeed | Mootori 2 soovitud pöörlemiskiirus |
| 32 | 16 | int16 | rpm | Motor3RequestSpeed | Mootori 3 soovitud pöörlemiskiirus |

Mootorid on eeldatud asetsema järgneva joonise alusel, kus joon keskel näitab roboti eeldatavat edasiliikumise suunda. Mootorite pöörlemise positiivne suund määratakse parema käe reegliga, kus mootorist ümbert kinni võttes pöial osutab ratta poole ning ülejäänud sõrmed näitavad positiivset pöörlemise suunda. Negatiivne pöörlemise suund on vastupidine positiivsele.



Mootorite kiiruse sõnumis on kolm 16-bitist signaali, mis tähistavad mootori (mitte ratta) pöörlemiskiirust väärtustes -32768 kuni +32767. Pluss tähistab positiivset pöörlemiskiirust, miinus tähistab negatiivset pöörlemiskiirust. Reaalselt saavutatav kiirus on loomulikult madalam ja see peaks olema tarkvaras konstandina kirjas.

## 0xD1 – Motor1Status, 0xD2 – Motor2Status ja 0xD3 – Motor3Status

Mootori staatuse kirjeldamise sõnum. Kõigis sõnumites on signaalidel samad nimed, aga mootorite indeksid on kas 1, 2 või 3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 16 | int16 | rpm | Motor1ActualSpeed | Mootori 1 reaalne pöörlemiskiirus |
| 16 | 16 | Int16 | - | Motor1EncoderClicks | Mootori 1 enkoodri muutus viimasest sõnumist. |
| 32 | 8 | uint8 | 100 mA | Motor1CurrentDraw | Mootori 1 voolutarve |
| 40 | 8 | uint8 | °C | Motor1DriverTemp | Mootorikontrolleri 1 temperatuur |
| 48 | 1 | bool | T/F | Motor1BridgeAFault | Mootorikontrolleri 1 poolsilla A viga |
| 49 | 1 | bool | T/F | Motor1BridgeBFault | Mootorikontrolleri 1 poolsilla B viga |
| 50 | 6 | - | - | - | Reserveeritud diagnostikaks |

Kolm mootorite olekusõnumit on samasuguse ülesehitusega. Sõnumite signaalid annavad infot mootorite töö kohta. Voolutarve on 100 mA ühikutes, ehk binaarväärtusele 35 vastaks 3,5A.

## 0xD4 - MotorPID

Mootoritele PID konstantide P, I ja D seadmine.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 32 | uint32 | - | ConstantAddress | STM32 kontrolleri mälu aadress |
| 32 | 32 | uint32 | - | ConstantValue | P, I või D väärtus |

## 0xB0 – BallControl

Pallimooduli juhtsignaalid.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 1 | bool | T/F | BallCharge | Kondesaatori laadimise lubamine. |
| 1 | 7 | uint8 | % | BallDribblerPower | Rullimootori PWM tööaeg. |
| 8 | 10 | uint16 | ms | BallKickTime | Solenoidi löögiaeg. |
| 18 | 1 | bool | T/F | BallKick | Solenoid löömine. |

*BallCharge* kaudu juhitakse kondensaatori laadimisskeemi tööd. Tõese väärtus (1) puhul on automaatne laadimine lubatud, väära (0) puhul mitte. Kui laadimine keelatakse, siis kondensaatori laadimine peatatakse, kuid seda ei tühjendata.

*BallKick* kaudu antakse käsk palli löömiseks solenoidiga. Tõese väärtus (1) puhul on palli löömiskäsk antud, väära (0) puhul mitte. Kui väärtus jätta tõeseks, siis tehakse löök automaatselt kondensaatori laadimise lõppedes. Seega ühekordseks löögiks peab väärtuse enne laadimise lõppemist ära nullima.

*BallKickTime* kaudu määratakse solenoidi transistori lahtioleku aeg. See mõjutab löögi aega ja tugevust. Väärtus on 0-1023 millisekundit.

*BallDribblerPower* määrab rulli mootori töötsükli aja protsentides. Väärtuse 0 puhul mootor seisab, väärtuse 100 (või suurema) puhul töötab maksimaalse võimsusega.

## 0xB1 – BallStatus

Pallimooduli olekusignaalid.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 1 | bool | T/F | BallContact | Rullianduri olek (kontakt palliga). |
| 1 | 1 | bool | T/F | BallInSight | Optilise palli anduri olek. |
| 2 | 1 | bool | T/F | BallCapacitorCharged | Kondensaatori laetuse olek. |

*BallContact* näitab kas pall (või muu objekt) on rulli all. Tõene väärtus (1) puhul on rull tõusnud, 0 (False) puhul mitte.

*BallInSight* on optilise pallianduri olek. Optiline andur peaks nägema palli mõned sentimeetrid enne rulli. Tõene väärtus (1) näitab, et palliandur näeb palli või muud objekti, väär (0) vastupidist.

*BallCapacitorCharged* on löögielektroonika kondensaatori laetuse olek. Tõese väärtuse (1) puhul on kondensaator täielikult laetud, väära (0) puhul (veel) mitte.

## 0xB2 – LineStatus

Jooneandurite olekusignaalid.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 8 | uint8 | % | LineLeftIntensity | Vasaku jooneanduri intensiivsus. |
| 8 | 8 | uint8 | % | LineRightIntensity | Parema jooneanduri intensiivsus. |

*LineLeftIntensity* ja *LineRightIntensity* näitavad optilistesse peegeldusanduritesse tagasipeegeldunud valguse intensiivust. Väärtuse 0% puhul on valguse intensiivsus minimaalne (optotransistor on suletud), väärtuse 100% puhul on valguse intensiivsus maksimaalne (optotransistor on täielikult avatud).

## 0xC0 – UIControls

Juhtpaneeli juhtliideste olekud.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 1 | bool | T/F | ButtonTopLeft | Ülemise vasaku nupu olek. |
| 1 | 1 | bool | T/F | ButtonTopRight | Ülemise parema nupu olek. |
| 2 | 1 | bool | T/F | ButtonBottomLeft | Alumise vasaku nupu olek. |
| 3 | 1 | bool | T/F | ButtonBottomRight | Alumise parema nupu olek. |

*ButtonTopLeft*, *ButtonTopRight*, *ButtonBottomLeft* ja *ButtonBottomRight* on kommunikatsioonimooduli paneelil olevate nuppude olekud. Tõese väärtuse (1) puhul on nupp alla vajutatud, väära (0) puhul mitte.

## 0xC1 – UIFeedback

Juhtpaneeli tagasisidefunktsioonide käsud.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 8 | uint8 | ms | LEDRed | Punase valgusdioodi põlemise aeg. |
| 8 | 8 | uint8 | ms | LEDGreen | Rohelise valgusdioodi põlemise aeg. |
| 16 | 16 | uint16 | Hz | BeepFrequency | Piiksu sagedus. |
| 32 | 8 | uint8 | ms | BeepDuration | Piiksu pikkus. |

*LEDRed* ja *LEDGreen* signaalid juhivad vastavalt punase ja rohelise valgusdioodi põlemise aega. Väärtuste 1 ms ja suuremate puhul põleb valgusdiood vastava aja alates sõnumi kättesaamisest. Väärtuse 0 puhul valgusdioodi ei juhita, ehk sellega saab lasta eelmisel valgusdioodi käsul edasi toimida.

*BeepFrequency* määrab ära soovitud helisignaali sageduse. Väärtus on 0 kuni 65535 hz, kuid loomulikult sõltub saavutatav helikõrgus kõlarist. Reaalne väärtus on 500 hz kuni 5000 hz.

*BeepDuration* määrab kas teha helisignaal (piiks) ja kui pikk. Väärtuse 1 ms ja suuremate väärtuste puhul kostab signaal vastava aja alates sõnumi saabumisest. Väärtuse 0 puhul heli ei tehta ja kui sõnumi saabumisel pole eelmine helisignaal lõppenud, siis seda ei peatata kah. Kui eelmine helisignaal pole lõppenud ja tuleb käsk uue jaoks, siis alustatakse koheselt uue signaaliga.

NB. Aku tühjenemise korral võib juhtmoodul helisignaali käskusid ignoreerida ja kasutada automaatset hoiatussignaali.

## 0xC2 – UIScreenClear

Juhtpaneeli ekraanil rea kustutamise sõnum. Kokku on ekraanil 4 rida.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 3 | uint8 | - | ScreenRow | Ekraani rea number. |

*ScreenRow* näitab rida, mis tekstist kustutada. Väärtuste 0 kuni 3 puhul kustutatakse rida 1 kuni 4. Ridasid loetakse ülevalt alla. Väärtuse 4 kuni 7 korral kustutatakse kogu ekraan.

## 0xC3 – UIScreenWrite

Juhtpaneeli ekraanile teksti kirjutamise sõnum. Ekraanile mahub 4 rida 16-märgist ASCII kodeeringus teksti. Teksti saab vanast tekstist üle kirjutada.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 4 | uint8 | - | ScreenRow | Ekraani rea number. |
| 4 | 4 | uint8 | - | ScreenColumn | Ekraani tulba number. |
| 8 | 4 | uint8 | - | ScreenTextLen | Teksti pikkus mida kirjutada (1-6) |
| 16 | 48 | char |  | ScreenText | Tekst (6 märki) |

*ScreenRow* näitab rida millele tekst kirjutada. Väärtused 0 kuni 3 tähistavad ridu 1 kuni 4. Ridasid loetakse ülevalt alla.

*ScreenColumn* näitab tulpa millest alates teksti kirjutada. Tulpasid loetakse vasakult.

*ScreenTextLen* näitab mitu märki sõnumis olevast tekstist ekraanile kirjutada. Kuna sõnumi pikkus on deterministlikusse säilitamiseks fikseeritud, siis pikkust määrab see signaal. Väärtuse 0 puhul kirjutatakse ekraanile üks (esimene) märk, väärtuse 5 (või suurem) puhul kõik 6 märki.

*ScreenText* sisaldab ASCII kodeeringus 6-märgilist teksti. Kui kirjutatada soovitakse vähem kui 6 märki, siis ülejäänud märgid võivad olla 0xFF. Seda kas kasutada saab kogu ASCII tabelit, sõltub kommunikatsioonimoodulist. Suur ja väike ladina tähestik, numbrid ja kirjavahemärgid peaksid kindlasti toetatud olema.

## 0xA0 - PowerStatus

Aku seisundi olekusõnum.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 16 | uint16 | mV | Battery1Voltage | Aku 1 (mootorite aku) pinge. |

*Battery1Voltage* näitab aku 100 ms jooksul keskmistatud pinget. Väärtused on 0 kuni 65535 mV.

## 0xF0 – Sensors

Sensorite näitude sõnum.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bitt | Pikkus | Tüüp | Ühik | Signaali nimi | Selgitus |
| 0 | 8 | uint8 | cm | DistanceSensor1 | Distantsisensori 1 kaugusnäit. |
| 8 | 8 | uint8 | cm | DistanceSensor2 | Distantsisensori 2 kaugusnäit. |
| 16 | 8 | uint8 | cm | DistanceSensor3 | Distantsisensori 3 kaugusnäit. |
| 24 | 8 | uint8 | cm | DistanceSensor4 | Distantsisensori 4 kaugusnäit. |
| 32 | 8 | uint8 | cm | DistanceSensor5 | Distantsisensori 5 kaugusnäit. |
| 40 | 8 | uint8 | cm | DistanceSensor6 | Distantsisensori 6 kaugusnäit. |
| 48 | 6 | uint8 |  | BeaconReceiver | Majaka vastuvõtja |

*Distancesensor* signaal näitab kaugust sentimeetrites.

*BeaconReceiver* signaal on võimeline edastama kuni kuuesektorilise majaka vastuvõtja tulemust. Robot on jagatud sektoriteks ning igale sektorile on signaalis omistatud oma bitt (vt. allpool olevaid jooniseid), kus binaarväätus 1 näitab, et majaka signaal on vastuvõtjasse jõudnund ning väärtus 0, et majaka signaali ei ole.

