Andri Raat

ÜKSIKKORRAS VALMISTATUD SÕIDUKI KIIRUSSIGNAALI TÖÖTLUS JA JUHTARVUTI MODIFITSEERIMINE

LÕPUTÖÖ

Tallinn 2014

Andri Raat

ÜKSIKKORRAS VALMISTATUD SÕIDUKI KIIRUSSIGNAALI TÖÖTLUS JA JUHTARVUTI MODIFITSEERIMINE

LÕPUTÖÖ

Transporditeaduskond

Autotehnika eriala

Tallinn 2014

SISUKORD

SISSEJUHATUS

Ottomootori tööpõhimõte pole viimase saja aasta jooksul muutunud, kuid kaasaegsel autol pole oma esivanematega suurt midagi pistmist. Mootorid on muutunud üha efektiivsemaks ning keskkonnasõbralikumaks. Selle saavutamisele on kaasa aidanud kindlasti elektrooniliste kontrollsüsteemide kasutuselevõtt. Samuti on need süsteemid muutunud üha keerukamaks ja täpsemaks. Elektroonika on arenenud eksponentsiaalse kiirusega [1]. Praegu võib ühe ruutsentimeetri suurusesse mikrokiipi paigutada rohkem arvutusvõimsust kui paarkümmend aastat tagasi tipp superarvutisse. Elektroonikakomponentide saadavus ja valik on samuti märgatavalt paranenud.

Antud lõputöö autoril on valmimas on üksikorras valmistatud sõiduk, mille aluseks on võetud Lotus 7. Sõidukil kastutakse BMW M50B25 mootorit ning sama mootori juhtimissüsteemi, juhtmestikku ning näidikuteplokki. Sõiduki šassii on kohandatud mahutamaks suuremat mootorit ning konstrueerimise lihtsustamiseks ja ohutuse tagamiseks on paljud kasutatavad komponendid pärit seeriatootmises olevatelt sõidukitelt.

Enamik komponente on valitud tagamaks korrektse töö oma põhifunktsioonis, seetõttu on kaotatud mõni lisafunktsioon. Näiteks kasutas BMW kiirussignaali lugemiseks keelreleed, mis oli paigutatud diferentsiaalikorpusesse. Kasutusel on aga Ford Sierra diferentsiaalülekanne millel puudub kiiruse lugemise võimalus.

Sõiduk on projekteeritud kasutamiseks nii tänavaliikluses kui ka rajal. Need kaks keskkonda aga erinevad kardinaalselt ja vajavad erinevaid seadistusi. Näiteks kasutatavad rattad peavad rajal tagama hea juhitavuse. Tänavaliikluses, kus kiirused on väiksemad, võib rõhku panna ka väljanägemisele. Kasutatavad rattad rajal on mõõtudega 195/60 R15, kuid tänaval 245/40 R18. Samuti peaks rajale minnes saama muuta mootori väljundkarakteristikat. Tähtis on suurematel pööretel väändemoment ning maksimumvõimsus, kuid erinevalt tänavaliikluses ei pea kütusekulule nii palju tähelepanu pöörama. Enamik moodsaid sõidukeid omab samuti mitut erinevat töörežiimi, mida juht saab vajadusel vahetada. Näiteks Alfa Romeo on kasutusel DNA süsteem, mis muudab mitmeid parameetreid, kaasa arvatud mootori väljundkarakteristikat [2].

Neid kriteeriumeid arvestades, on käesoleva lõputöö eesmärgiks leida lahendus kiirusesignaali saamiseks kasutatavasse näidikuteplokki, kalibreerida signaal vastavalt kasutatavatele ratastele ning muuta vastavalt vajadusele mootori väljundkarakteristikat kasutades originaal juhtarvutit.

Originaal juhtarvuti kasutamise eeliseks programmeeritavate juhtarvutite ees on kasutajamugavus. Kuna tehase poolt on seadistatud mootor töötama kõikvõimalikes tingimustes(mägedes, merepinnal, miinuskraadidel, erineva kvaliteedi ja oktaaniarvuga kütust kasutades jne), siis ei pea muretsema näiteks külmkäivituste pärast. Programmeeritavat juhtarvutit kasutades (eriti odavamaid) võib väita, et seadeid igasugustele tingimustele üks inimene teha ei suuda. Universaalsuse arvelt aga on võimalik optimeerida mootori tööd ning sellega saavutada lisavõimsus.

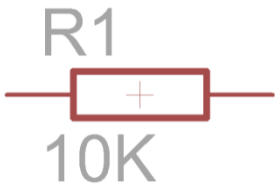
1. ELEKTROONIKAKOMPONENDID JA –SKEEMID

Elektroonikakomponentide mõistmine on aluseks elektriskeemide tööpõhimõtte välja selgitamiseks. Antud peatükis kirjeldatakse tähtsamaid ja töös kasutatavaid komponente.

1.1 Takistid

Takistid(Foto 1)(Joonis 1) on kõige rohkem kasutatavad komponendid elektriskeemides. Takisteid iseloomustab peamiselt kaks parameetrit, milleks on takistus (oomi) ning võimsus (vatti). Takisteid kasutatakse peamiselt selleks, et piirata voolutugevust või tekitada kindlaid pingelange. Enamik takisteid on valmistatud peenikestest süsinikvarrastest. Süsinikvarda suurus määrab takisti takistuse. Võimsad takistid on tavaliselt valmistatud keritud traadist. [3, p. 18]

Foto 1.Erinevad takistid [4]



Joonis 1.Üks võimalik takisti sümbol Eagle *CAD(Computer-aided design)* paketis

1.2 Kondensaatorid

Kondensaatorid(Foto 2)(Joonis 2) on komponendid, mis salvestavad elektrilaengu. Kõige lihtsam kondensaator koosneb kahest plaadist, mis on omavahel eraldatud isolaatoriga. Mahtuvuse määrab plaatide pindala, nende vahekaugus ja isolaatori dielektriline läbivus. Isolaatorina võidakse kasutada näiteks paberit, mis on paigutatud kahe fooliumlehe vahele, mis omakorda pressitakse rullikeeratuna metallist korpusesse. Saavutamaks suuremaid mahtuvusi, tuleks isolaatorikiht teha võimalikult õhuke. Seda saab teha, kui tekitada ühe plaadi peale oksiidikiht (enamasti 10^{-4}mm õhuke). Niisugune plaatide töötlemisviis annab suurema mahtuvuse, kuid kondensaator muutub polaarseks ja talub ainult madalaid pingeid. Kondensaatorit iseloomustavad parameetrid on mahtuvus (farad) ja pingetaluvus (volti). [3, p. 18;20]

Foto 2. Erinevad kondensaatorid [4]

Joonis 2. Üks võimalik kondensaatori sümbol Eagle *CAD* paketis

1.3 Diood

Dioodi(Foto 3)(Joonis 3) võib enamasti vaadelda kui ühesuunalist klappi. Diood koosneb PN-siirdest, mis lubab elektronidel liikuda N-tüüpi materjalilt (pooljuht negatiivse pingega) P-tüüpi materjalile (pooljuht positiivse pingega) [3, p. 20]. Pooljuhtmaterjalideks kasutatakse nii germaaniumi kui seleeni, kuid kõige levinum on räni [5]. Ükski diood pole ideaalne. Kui vool läbib dioodi peab pinge olema kõrgem kui lävepinge. Enamasti jääb pingelang võrdseks lävepingega. Ränidioodidel on lävepinge 0,6-0,7V, germaaniumdioodidel 0,3V [6].

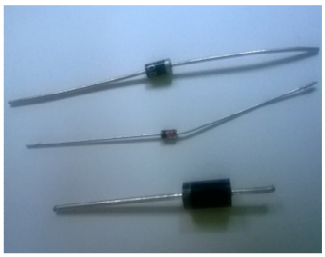
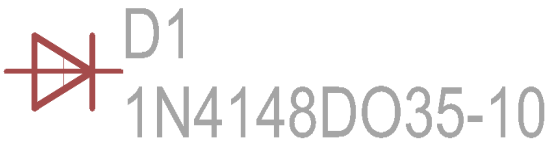


Foto 3. Erinevad dioodid [4]



Joonis 3. Üks võimalik dioodi tähis Eagle *CAD* paketis koos tüübi nimetuse ja kestaga

1.4 Transistor

Transistorid(Foto 4) on elektroonikakomponendid, mis võimaldavad valmistada tänapäevaseid keerukaid elektroonikasüsteeme. Transistorit saab kasutada nii lülitina kui ka võimendina. Valmistatud on nad samadest pooljuhtmaterjalidest nagu dioodidki ja kasutatakse kahte konfiguratsiooni NPN ning PNP. Transistoril on vähemalt kolm jalga – baas, kollektor ja emitter. Kui baasile rakendada väike vool, hakkab kollektori emitteri vaheline osa voolu juhtima Baasile rakendatav vool võib olla sadades kordades väiksem, kui kollektori emitteri vaheline vool. [3, p. 20]

Erinevalt tavalistest transistoritest on väljatransistoritel(inglise k *FET-Field Effect Transistor*) kõrge sisendtakistus. Jalgu kutsutakse sellisel transistoril lätteks(inglise k *source*), neeluks(inglise k *drain*) ning paisuks(inglise k *gate*). Kui pasiule rakendatakse pinge, hakkab lätte ja neelu vaheline osa juhtima voolu. [3, p. 20]

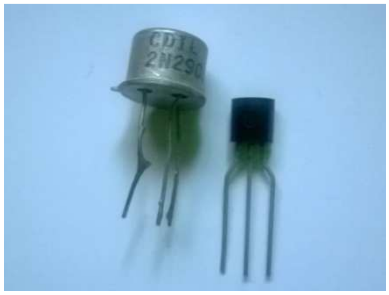
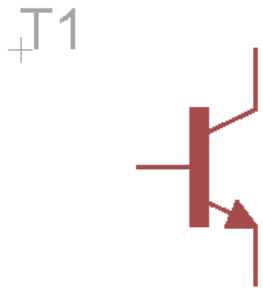


Foto 4. Erinevad transistorid [4]



Joonis 4. NPN transistori sümbol Eagle *CAD* paketis

1.5 Võimendid

Kõige lihtsam võimendi on võimalik valmistada vaid ühest transistorist ning takistist. Tavaliselt aga kasutatakse sellist võimendit rohkem lülitusteks, kui võimendamiseks. Sellise võimendi väljundkarakteristika ei pruugi aga tihti olla piisav, seetõttu on välja töötatud palju erinevaid võimendite tüüpe. Üheks nendest on operatsioonivõimendid, mida on võimalik kasutada signaalivõimendusteks ning ka puhvrina sensori ja koormuse vahel. Selliste võimendite siseehitus võib olla üpriski keerukas, kuid väliselt on neid lihtne juhtida. [3, p. 21]

Eritüüpi võimenditeks on näiteks komparaatorid, mis võrdlevad kahte elektrilist pinget. Põhimõtteliselt on komparaatoris ühendatud omavahel operatsioonivõimendi ning loogikalüli, sest väljud on digitaalne (kaks loogikanivood – loogiline 1=V+, loogiline 0=V-). [7]

1.6 Mikrokiibid

Mikrokiibid koostatakse õhukesele pooljuhtmaterjalist plaadile. Nad võivad sisaldada kõiki eelpool nimetatud komponente ning teostada erinevaid ülesandeid, nagu näiteks ümberlülitused, võimendused ja loogilised funktsioonid. Kuna mikrokiipides kasutatavad komponendid on väga väiksed ja asuvad üksteisele lähedal, on võimalik saavutada väga suuri lülituskiirusi (üle 1 MHz on tüüpiline). Mikrokiipe on saadaval väga erinevate funktsioonide täitmiseks ning erinevates korpustes. [3, p. 20;21]

Foto 5. Erinevad mikrokiibid [4]

1.7 Süsteemide arendusplatvormid.

Süsteemide arendusplatvormid on platvormid, millel on lihtne ja mugav prototüüplahendust testida. Tavaliselt on kõik komponendid lihtsasti ligipääsetavad ning sisenditele ja väljunditele on lisatud väljavõtted, et signaale mõõta ja kontrollida. Tarkvara kirjutatakse tavaliselt kõrge tasemega programmeerimiskeeles, näiteks C++ ja teisendatakse seejärel kompilaatori abil masinkoodi. Programmi laadimiseks arendusplatvormile või parameetrite kuvamiseks arvutis on enamasti kasutusel jadaühendus, näiteks RS232. [8, p. 46;47]

Antud töös kasutatakse Arduino Leonardo nimelist arendusplatvormi

1.8 Arduino mikrokontrolleri arendusplatvorm

Arduino arendusplatvorm (Foto 6) on suhteliselt lihtne kasutada ning vägagi paindlik. Protsessorina kasutab see Atmega32U4 mikrokontrollerit, millel on 20 digitaalset sisendit või väljundit ning millest 12 on võimalik kasutada analoog sisenditena. Arendusplaadile on integreeritud kõik protsessori tööks vajalik – sinna hulka 16MHz kristall, reguleeritud ja stabiliseeritud toiteplokk, USB andmeside, mille abil tekitatakse virtuaalne RS232 liides ja reset nupp. [9]

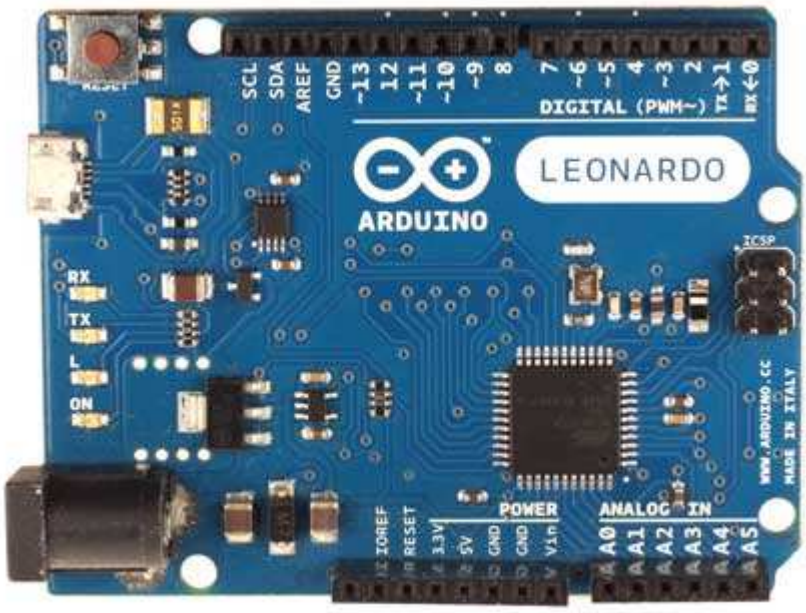


Foto 6. Arduino Leonardo mikrokontrolleri arendusplatvorm [9]

Atmega 32u4 on integreeritud USB kontroller, 32Kb *flash* mälu programmi jaoks ja 2,5 kb SRAM'i. Enamus instruktsioone võtavad aega ühe taktisageduse ning maksimaalne taktisagedus on 16 MHz. Lisaks on muutujate salvestamiseks 1Kb eeprom mälu. [10]

2. KIIRUSSIGNAALI SAAMINE NÄIDIKUTE PLOKKI

2.1 ABS anduri kasutamine kiiruse mõõtmiseks.

Töös kasutatav üksikkorras valmistatud sõiduk kasutab Ford Sierra diferentsiaal ja peaülekannet ning pooltelgi. Kuna BMW 525i-l asetseb kiirusandur diferentsiaali korpuses, siis kiirussignaali jaoks on vajalik leida muu võimalus. Kasutatavatele pooltelgedele on ABS (inglise k *antilock braking system*) hammasvööd integreeritud, siis ratta pöörlemiskiiruse lugemiseks on võimalik kasutada tagumise ratta ABS hammasvööd. See koosneb 80 hambast. Ratta kiiruse lugemiseks kasutatakse antud töös induktiiv ABS andurit. Sellise anduri väljundsignaali kuju on sinusoid, kusjuures signaali sagedus ja amplituud määravad ratta kiiruse. Hamba kuju ja sensori kaugus hambast määravad anduri tundlikkuse ning miinimum kiiruse, mille juures signaal loetav on [11, p. 60;61]. [3, p. 38]

2.2 Võimalusi kiirussignaaliks

Nagu eelpool mainitud on kasutatava induktiivanduri väljundsignaal sinusoid. BMW kasutas aga diferentsiaalist kiiruse lugemiseks keelreleed, mille väljundsignaal on nelinurklaine. Samuti erineb signaaliimpulsside arv ühe ratta täispöörde sooritamisel – ABS hammasvööl on neid 80, keelreleel aga üheksa hammast. Kõige lihtsam variant oleks modifitseerida (freesida) ABS võru sarnaseks BMW diferentsiaalis kasutatava hammasvööga, kuid niiviisi oleks siiski vajalik teisendada induktiivanduri sinusoidaalne väljundsignaal fikseeritud piikväärtusega nelinurklaineks. Samuti ei oleks sellisel lahendusel paindlikust erinevate rattamõõtude kasutamisel.

2.3 Korrektsiooni vajalikkus

Hetkel kasutatakse sõidukil rehve mõõtudega 195/60 R15, kuid tulevikus tänavaliikluses kasutades võivad rataste mõõdud muutuda. Sellise rehvi kõrguse laiuse suhe on arvutatav valemiga [12, p. 734]

$$\frac{H}{W_{suhe}} = \frac{H}{W} * 100,$$

(1)

kus H (mm) - rehvi ristlõike kõrgus;

W -rehvi ristlõike laius.

Sellest valemist (1) saame tuletada rehvi ristlõike kõrguse, milleks on

$$H = \left(\frac{H}{W_{suhe}} /100\right)$$

$$195 * \frac{60}{100} = 117\ (mm).$$

Kasutades ringi ümbermõõdu valemit

$$U = 2 * \pi * \mathrm{r},$$

(2)

kus U -ringi ümermõõt;

r -ringi raadius,

on sellise ratta ümbermõõt

$$2 * \pi * \left(117 + 15 * \frac{25{,}4}{2}\right) \approx 1923\ (mm).$$

245/40 R18 ümbermõõduks oleks aga:

$$2 * \pi * \left(245 * \frac{40}{100} + 18 * \frac{25{,}4}{2}\right) \approx 2052\ (mm).$$

Kui selliste mõõtudega ratas teeks 1000p/min ehk 60000p/h oleks kiiruseks

$$60000 * 2052 * 10^{-6} = 123{,}12\ (km/h).$$

Kui aga valitakse väiksema ümbermõõduga ratas oleks kiirus

$$60000 * 1923 * 10^{-6} = 115{,}38\ (km/h).$$

Nagu näha, siis nii erinevate mõõtudega rehve ja velgi kasutades peaks olema võimalus korrigeerida spidomeetri näitu vastavalt ratastele. Reaalsete kiiruste erinevus on üle 6%. Kuna mikrokontrollerid on tänapäeval väga laialdaselt levinud, odavad ja kompilaatorid on kasutajasõbralikud, siis kõige paindlikum variant on kasutada just neid.

BMW 525i mootori pöörete piiraja on seadistatud 6450 p/min. Viienda ehk kiireima käigu ülekandearv on 1,0 seega kardaan pöörleb täiskiirusel sama kiiresti kui mootor. Peaülekande ülekandearv on 3,14. Täiskiirusel teeb seega ratas

$$\frac{6450}{3{,}14} \approx 2054\ (p/min)$$

ehk 34 p/sek. Kui ABS hammasvööl on 80 hammast, siis maksimaalne signaali sagedus on

$$34 * 80 = \ 2730\ (hz)$$

ehk umbes 2,7Khz. Kiirus väiksema rattaga oleks sellisel juhul

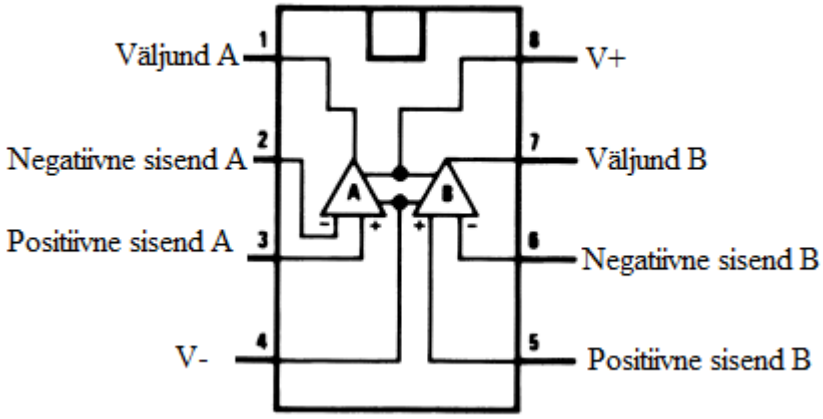
$$2054 * 60 * 1923 * 10^{-6} = 236{,}99\ (km/h)$$

ning suurema rattaga

$$2054 * 60 * 2052 * 10^{-6} = 252{,}88\ (km/h).$$

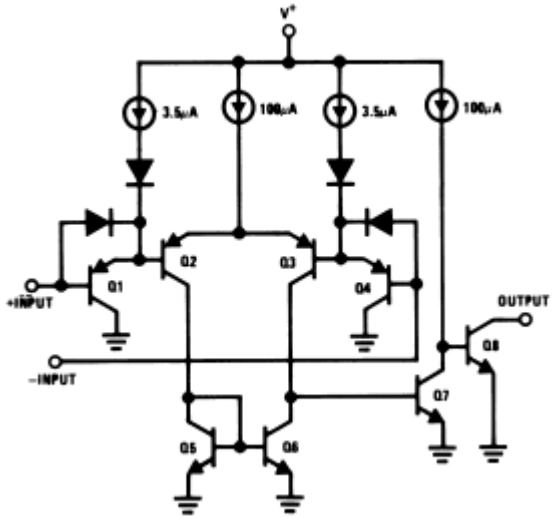
2.4 LM393 komparaator

ABS anduri sinusoidsignaali teisendamine digitaalseks nelinurksignaaliks peab toimuma mikrokontrolleri väliselt eraldiseisvas plokis. Selleks kasutatakse LM393 komparaatorit. See mikrokiip on turul olnud juba kaua aega, tõestanud oma vastupidavust, on kergesti kaubandusest hangitav ja tema näitajad on piisavad järgnevaks lahenduseks. LM393 sisaldab endas kahte identset komparaatorit (Joonis 5). [13]



Joonis 5. Pealtvaade LM393 komparaatorist [13]

LM393 komparaator vajab tööks positiivset ja negatiivset toitepinget. Ühel komparaatoril on kaks sisendit ja üks väljund. Väljund on avatud kollektoriga (inglise k *Open collector*) (Joonis 6). Kui negatiivse sisendi pinge on madalam, kui positiivse sisendi pinge on väljund ühendatud negatiivse toitepingega ning kui negatiivse sisendi pinge on kõrgem, kui positiivse sisendi pinge, siis on väljund skeemist lahti ühendatud.

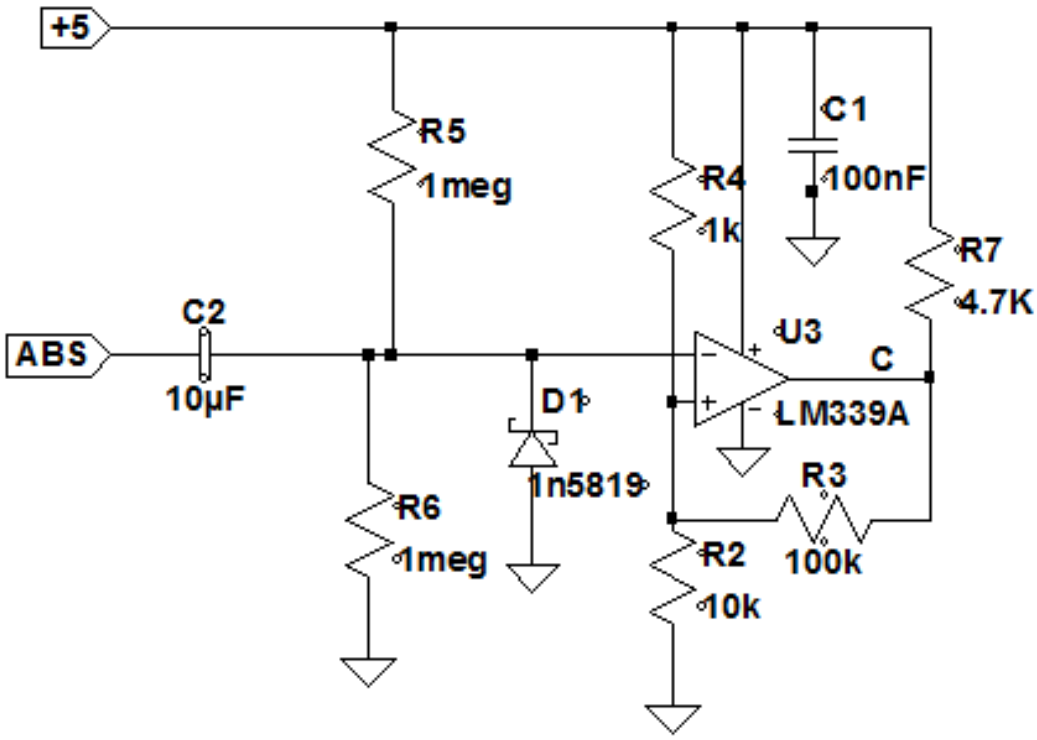


Joonis 6. LM393 ühe komparaatori siseskeem [13]

Mikroskeemi toitepinge peab jääma vahemikku 2-30V. Mikrokontrolleri arendusplatvormi toiteplokki on sisse ehitatud 5 V toitepinge(kuni 500mA) ning komparaatori väljundsignaal peab olema samuti 0-5V, siis kõige mugavam on kasutada just seda toitepinget. Mikroskeemi tööd toitepinge oluliselt ei muuda. Sisendpinged võivad olla kõrgemad, kui toitepinge, kuid mitte üle 36V. [14]

Neid parameetreid arvesse võttes tuleb konstrueerida vajalik riistvara.

2.5 Riistvara



Joonis 7. ABS induktiivanduri signaali teisendamine digitaalseks

Signaaliteisenduseks vajalik riistvara (Joonis7) koosneb kuuest takistist, kahest kondensaatorist LM393 mikroskeemist ning ühest *Schottky* dioodist. Auto maksimumkiirusel on kasutatava ABS anduri (osakoodiga 8E0927803A) väljundpinge absoluutväärtus 20V (väljundpinge sõltus pöörlemiskiirusest ning anduri kaugusest hammasvööst), seega mikroskeemi sisendpinge jääb lubatud piiridesse.

R2(10kΩ) ja R4(10kΩ) takistid moodustavad pingejagaja. Selle väljund peaks olema pool toitepingest, et sisendi madala pinge korral skeem tööle hakkaks. Pingejaga väljundpinge (keskpunktis olev pinge) on arvutatav valemiga [15]

$$V_{väljund} = \frac{V_{toide} * R_2}{(R_1 + R_2)}, \quad (3)$$

kus $V_{väljund}$[V] -pingejagaja väljund;

V_{toide}[V] -pingejagaja sisend;

R_1[Ω] -esimese takisti väärtus;

R_2[Ω] -teise takisti väärtus.

Seega pinge LM393 positiivses sisendis on

$$V_{väljund} = \frac{5 * 10 * 10^3}{(10 * 10^3 + 10 * 10^3)} = 2{,}5(V).$$

R5(1MΩ) ja R6(1MΩ) moodustavad samasuguse pingejagaja, et tõsta ABS anduri potentsiaal väikese sisendpinge korral 2,5V kõrgemale negatiivsest toitepingest. Seda pingejagajat võib vaadelda ka kui virtuaalset maandust ABSi anduri signaalile.

Takisti R7(4,7kΩ) toimib tõmbe (inglise k *pull-up*) takistina, kuna LM393 komparaatoril on avatud kollektoriga väljund. Kui väljundtransistori (Joonis 6) baasil pole pinget, võib sama transistori kollektorit antud situatsioonis vaadelda kui lahtiühendatud klemmi. Takisti ühendamisel toitepinge ja komparaatori väljundi vahele „tõmbab“ see väljundi potentsiaali samale tasemele toitepingega(5V). Väärtus on valitud kasutades Ohmi seadust, et väljundis oleks vähemalt 1mA voolu.

$$U = I * R,$$

(4)

kus U[V] -pinge;

I[A] -voolutugevus;

R[Ω] -takistus.

$$R = \frac{U}{I} = \frac{5}{1 * 10^{-3}} = 5 * 10^3$$

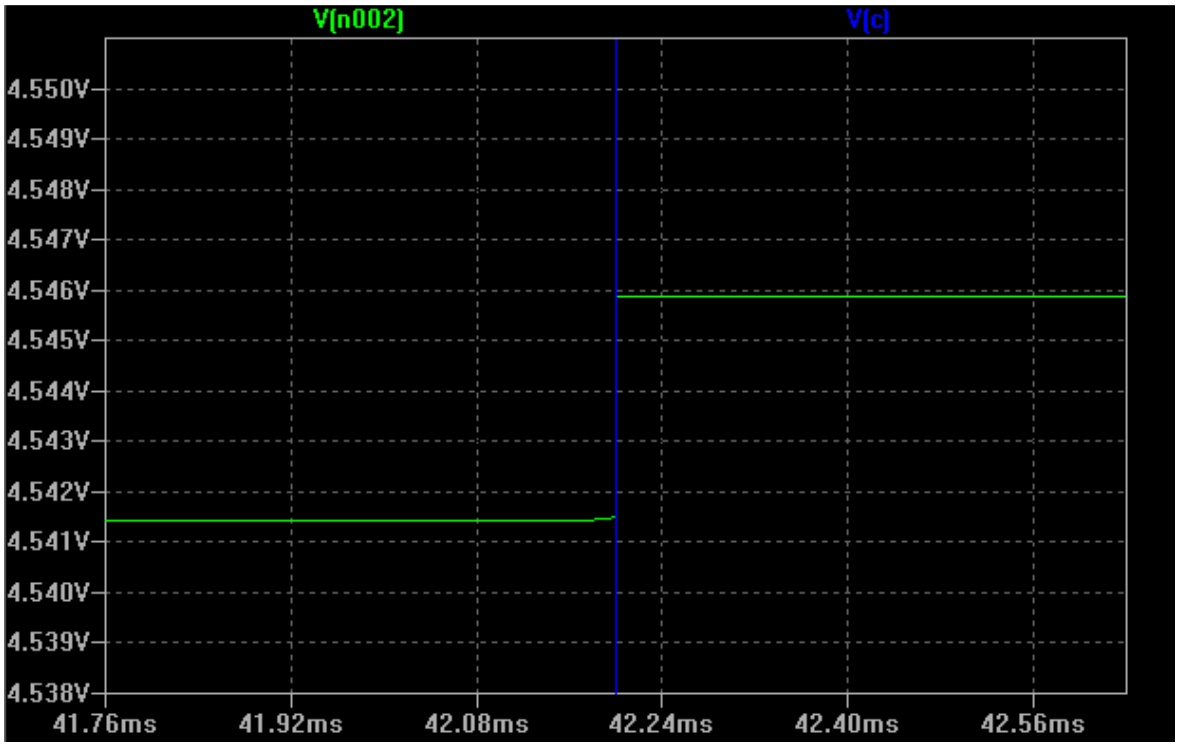
Takisti väärtus peab olema 5kΩ, seega tuleb takistite standardreast valida lähim väikseim väärtus milleks on 4,7kΩ [16].

Kondensaator C1(100nF) on toitepinge silumiseks. Kuna mikroskeem vajab erinevatel ajahetkedel erinevaid voolutugevusi, siis häirete vältimiseks ja silumiseks on toiteliinide vahele ühendatud 100nF suurune keraamiline kondensaator. Trükkplaadil peab mainitud kondensaator paiknema mikroskeemile võimalikult lähedal.

Kondensaator C2(10μF) on samuti häirete silumiseks, kuna juhtmed andurist skeemini võivad korjata kõrgsageduslikku müra. Seetõttu on kondensaatori mahtuvus valitud suurem. kui mikroskeemi toitepinge kondensaator.

Schottky diood D1 blokeerib negatiivse pinge, kuna komparaatori sisendid võivad langeda vaid 0.3V madalamale negatiivsest toitepingest. 1N5819 maksimaalne blokeerimispinge on 40V [17], mis on piisav, kuna ABS anduri absoluutpinge oli 20V. Samuti on sellel dioodil madal pingelang – 0,3V [17].

Takisti R3(100kΩ) on hüstereesi tekitamiseks. Takisti väärtus on valitud skeemi simuleerimise teel. Komparaator suudab väljundi olekut muuta mitme megahertsise sagedusega ja seega võib tekkida olukord, kus mõlemad sisendid on täpselt sama pingega ning väljund jääb mõneks hetkeks kahe oleku vahele pendeldama. Sellise olukorra vältimiseks tuleb muuta positiivse sisendi pinget vastavalt väljundi olekule. Piisav on umbes 1-10 mV [13]. Samal hetkel kui väljund muudab olekut madalast signaalist kõrgeks, muudetakse mõnevõrra R4 ja R2 pingejagaja suhet nii, et sisendpinge muutub umbes 5 mv võrra kõrgemaks ja kui väljund ühendatakse maaga, muutub see samavõrra väiksemaks (Joonis 8).



Joonis 8. Hüsterees: *sinine* – väljundsignaali tõusmine; *roheline* – positiivse sisendi pingemuutus

Simuleerimiseks kasutati LTSpice tarkvara versiooni 4.20. See on tasuta saadaval [18] ning väga suurte võimalustega. Virtuaalsete funktsioonigeneraatoritega genereeritakse vajalikud signaalid ning pinged, määratakse simulatsiooni tüüp(antud juhul ajas muutuv) ning seejärel on võimalik kasutada virtuaalset ostsillograafi või mõõta voolutugevusi. Vajadusel saab muuta komponentide väärtusi või neid välja vahetada sobivate vastu.

Simulatsiooni tulemused on välja toodud lisades(Lisa 1).

Komaraatorist väljuva nelinurklaine laius on küll muutuv vastavalt sisendsignaali amplituudile, kuid kuna mikrokontrolleri abil loetakse vaid tõusvaid signaaliservasid, siis ei oma see tähtsust

2.6 Tarkvara

Mikrokontrolleri tarkvara on kirjutatud kasutades kompilaatorit Arduino IDE 1.0.5 versiooni [19]. Programmeerimiskeel on väga sarnane C programmeerimiskeelega, kuid keelde on lisatud palju unikaalseid funktsioone programmeerimise lihtsustamiseks [20]. Näiteks fikseeritud sagedusega laiusimpulssmodulatsiooni (*PWM – PulseWidth Modulation*) signaali väljastamiseks piisab vaid ühest reast koodist, millega pannakse paika väljastav viik ning impulsside laius [21].

Programmi(Lisa 5) võib jagada kolme põhiossa:

- algväärtuste seadistamine;
- katkestuse töötlus;
- põhiosa.

Kõigepealt määratakse muutujad ning kasutatavad teegid, kusjuures heaks tavaks on konstandid kirjutada suurte tähtedega ja programmi töös muutuvad muutujad väikestega. Samuti kirjutatakse enamasti muutujad inglise keeles. Vajalikeks konstantideks on:

- *TREAD_WIDTH* – rehvi (mm);
- *ASPECT_RATIO* – rehvi kõrguse ja laiuse suhe;
- *RIM_SIZE* – velje läbimõõt (tolli);
- *ABS_TOTAL_TEETH* – pulsside arv ühe ratta täispöörde sooritamisel;
- *SPEED_OUT_PIN* – töödeldud kiirussignaali väljundviik;
- *SPEED_IN_INTERRUPT* – kiiruse sisendviik (LM393 väljund);
- *TIRE_CIRC* – arvutatud ratta ümbermõõt.

Programmisisesed muutujad on:

- *counter* - muutmälus asuv muutuja kus hoitakse loetud impulsside arvu;
- *prev_time* - aeg millal tehti viimane ratta täispööre;
- *current_time* - aeg millisekundites alates programmi käivitumisest;
- *time_delta* - aeg, kaua viimane ratta täispööre aega võttis;
- *velocity* - arvutatud kiirus ühikuks mm/ms;
- *vspeed* - väljund kiirusesignaali sagedus, kuid kuna 50hz juures näitab; spidomeeter 50km/h ning 100hz juures 100km/h siis võib seda vaadelda kui väljund kiirust km/h.

Muutujad, mis vajavad komakohti, on defineeritud kui ujukomaarvud ning nad salvestatakse 32bitiste muutujatena [22].

Muutujad, mille numbriline väärtus läheb aina suuremaks (näiteks *current_time)* on defineeritud kui *unsigned long* . Selline muutuja peab olema täisarv, kuid kuna see salvestatakse 32bitisena, siis maksimumväärtus saab olla 2^{32} [23].

Volatile märksõna kasutatakse kompilaatorile informatsiooni andmiseks. Väga kiiresti muutuvad muutujad paigutatakse sellisel juhul RAM mällu ja nendega manipuleerimine on kiirem [24].

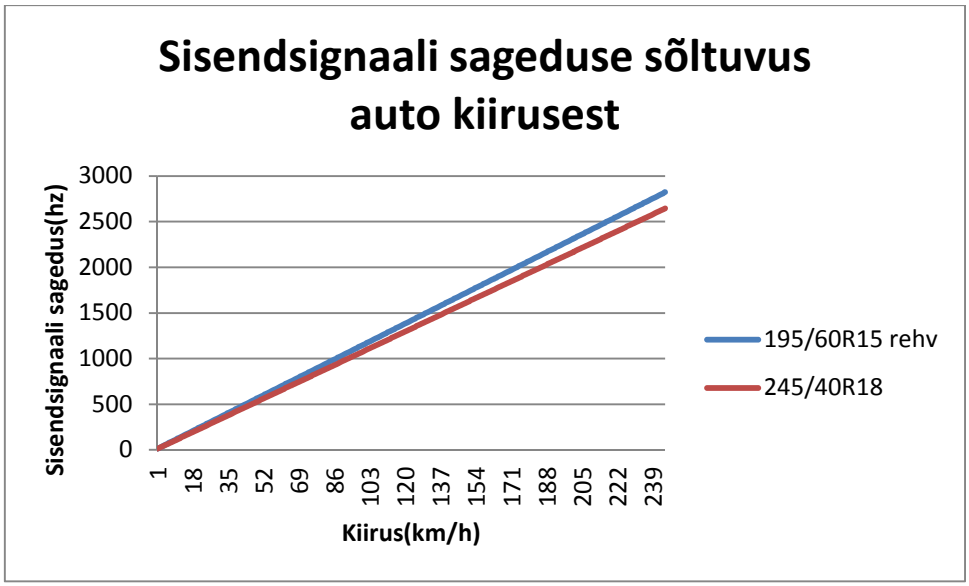
Algväärtuste seadistamise plokk (*setup*) läheb tööle iga kord, kui mikrokontroller käivitatakse (süüde sisse keeratakse) [25]. Seejärel konfigureeritakse töödeldud kiirussignaali väljundviik tööle väljundina, määratakse viimase rattapöörde sooritamise aeg (see on vajalik, et peale esimest rattapööret väljastatakse õige sagedus) ja seadistatakse kiiruse sisendviik aktiveerima katkestuse plokki tõusval signaaliserval.

Kui kiiruse sisendviigult avastatakse tõusev signaaliserv, aktiveeritakse katkestuseplokk(*count up*), mis suurendab loetud impulsside arvu ühe võrra. Samal ajal jäätakse ülejäänud programmi töö hetkeks pooleli ja jätkatakse, kui muutuja on suurendatud. Nii tagatakse, et kõik signaalimpulsid registreeritakse ja väljund on võimalikult täpne. [26]

Programmi põhiosas(*loop*) toimub hetkekiiruse ja väljundsageduse arvutamine. See programmiosa käivitatakse uuesti alati peale lõpuni jõudmist ja nii lõpmatuseni või kuni süüde välja keeratakse või registreeritakse katkestus. Kõigepealt kontrollitakse, kas muutuja *counter* on suurem või võrdne impulsside arvuga ühe ratta täispöörde sooritamisel. Kui ei ole, kontrollitakse uuesti ja kui on, arvutatakse hetkekiirus ning väljundsagedus. Kontrollida võiks ka ainult võrdsust, kuid kui katkestus registreeritakse kontrollimise hetkel võib juhtuda, et muutuja *counter* väärtus läheb suuremaks kui impulsside arv ratta täispöördel, seega tingimus muutub tõeseks alles pärast 16 bitise väärtuse maksimeerimist(*counter* on defineeritud kui *int)* [27] ja järgmine väljundi uuendus toimub peale 409 rattapööret.(32767/80≈409).

Kui kirjeldatud mitterange võrratus on tõene, võrdsustatakse muutuja *current_time* ajaga mil programm tööle hakkas [28]. Sedasi saadakse relatiivne hetkeaeg. Seejärel lahutatakse hetkeajast eelmine aeg, mil ratas täispöörde sooritas ja salvestatakse see muutujasse *time_delta.* See muutuja sisaldab aega, kui kaua võttis üks ratta täispööre millisekundites aega. Järgmiseks võrdsustatakse eelmine täispöörde aeg hetkeajaga, et hakata mõõtma aega kuni järgmise täispöörde sooritamiseni. Siis võrdsustatakse loetud impulsside arv nulliga ning arvutatakse kiirus. Ratta ümbermõõt jagatakse ratta täispöörde sooritamiseks kulunud ajaga. Kuna ratta ümbermõõt oli konstanti salvestatud millimeetritena ning aega mõõdetakse millisekundites, siis kiirust mõõdetakse ühikutes mm/ms.

Väljundsageduse arvutamiseks kasutatakse *map* funktsiooni [29], kuna kiiruse ning sisendsageduse vahel on lineaarne seos (Joonis 9)



Joonis 9. Sisendsignaali sageduse sõltuvus auto kiirusest

See funktsioon paigutab ühe numbrivahemiku lineaarselt teise numbrivahemikku ja väljastab teise numbrivahemiku väärtuse. Näiteks kui üheks numbrivahemikus on 0-100 ja teiseks 0-10, ning sisendarvuks on 40 siis funktsiooni väljund on 4. Kiiruse, mis asub muutujas *velocity*, ühikuteks oli mm/ms, kuid sama numbriline väärtus on ka ühikul m/s. Seega üheks numbrivahemikuks peab olema 0-70(m/s)(väärtus on reaalsest maksimumkiirusest valitud mõnevõrra suurem, et kogu pööretevahemik oleks kindlasti kaetud). Väljundsagedus moodustus selliselt, et 1 km/h vastab sagedus 1 Hz, siis teiseks numbrivahemikuks peab olema 0-252. Muutuja *velocity* väärtus on defineeritud kui ujukomaväärtus, kuid funktsioon töötab vaid täisarvudega, siis korrutatakse muutuja läbi sajaga, seejärel teisendatakse lähimaks täisarvuks ning ka esimese numbrivahemiku väärtused korrutatakse läbi sajaga. Selliselt ei kaotata ülemäära palju resolutsioonis.

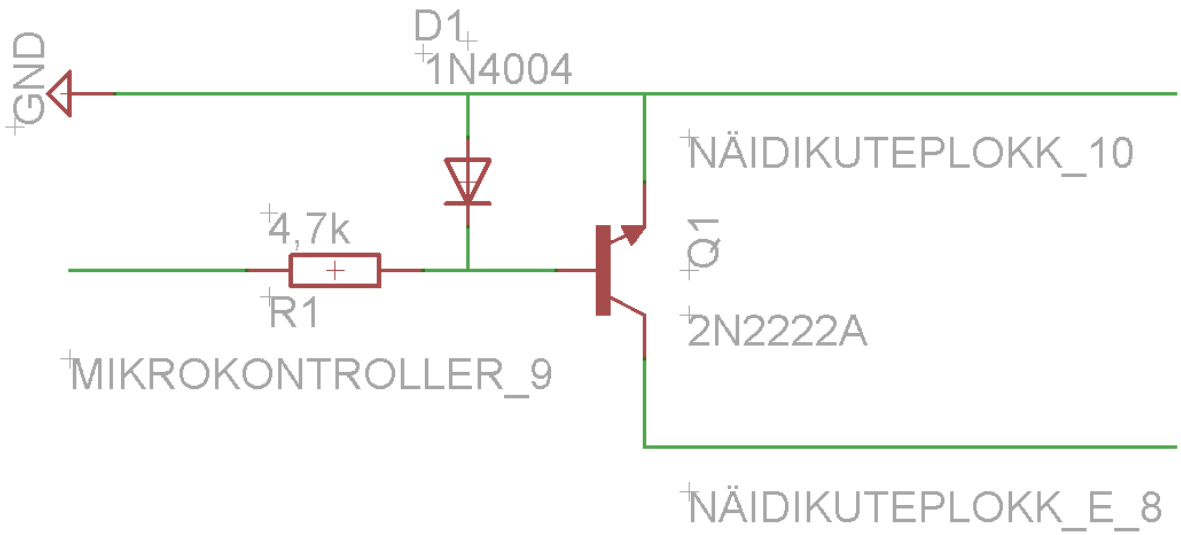
Väljundsageduse väljastamiseks kasutatakse eraldi teeki *toneAC* [30], mis võimaldab erinevalt sisseehitatud *tone* funktsioonist väljastada ka väga madalaid sagedusi(1Hz). Sisseehitatud *tone* funktsioon võimaldab väljastada signaali alates 31Hz [31]. Teek on põhiliselt mõeldud kasutamiseks helide väljastamiseks, kuid kuna genereeritakse vajalik 50% laiusega nelinurklaine siis sobib see kasutamiseks ka antud töös. Kõigepealt tuleb kompilaatorile anda käsk teegi kasutamiseks antud programmis*(#include <toneAC.h>)*, seejärel kasutades funktsiooni *toneAC()* saab väljastada eelarvutatud sageduse.

Väljundsignaali täpsus jääb ühe protsendi piiresse, mis on piisavalt täpne(Lisa 6-10). Tulevikus oleks võimalik mikrokontrollerile lisada positsioneerimisseade (inglise k *GPS* - Global Positioning

System) ja salvestada väärtused mälukaardile, et neid hiljem töödelda. Selline süsteem oleks kasulik ringiaegade analüüsimiseks.

2.7 Väljund

Mikrokontrollerist väljastatakse õige sagedusega kiirussignaal. Originaalis kasutatav keelrelee ühendub otse näidikuteplokki läbi pistiku E viikide 10 ja 8 [32](Lisa 11). Nende ühendamisel loetakse impulss. Viik number 10 on negatiivse ning kaheksa positiivse pinge jaoks. Keelrelee asendamisel transistoriga saab eelmainitud juhtmed omavahel ühendada ja lahutada.



Joonis 10. Ühendusskeem asendamaks keelreleed

Kasutatakse 2N2222A transistorit, mis on loodud kasutamiseks lülitusfunktsioonides ja on piisavalt võimas ning lihtsalt kättesaadav. Lülitada suudab kuni 0,8A voolu ning lubatud kollektori ja emitteri vaheline pinge on 40V [33].

Takisti R1 on mikrokontrolleri väljundi kaitseks, et piirata vool 1mA peale. Nii saab läbi juhtmete, mis lähevad näidikuteplokki, liikuda vähemalt 50mA. Kuna tegemist on digitaalse signaaliga, siis on voolutugevused madalad.

Diood D1 kaitseb transistorit negatiivsete pingepiikide eest kui neid peaks juhtuma. Kasutatud on ühte laialdasemalt kasutatavat dioodi 1n4004 [34].

3. JUHTARVUTI PROGRAMMI MUUTMINE

Selles peatükis kirjeldatakse mootori juhtimissüsteemi, programmimälu ning selle tööpõhimõtet. Samuti konstrueeritakse elektriskeem, mille abil mäluplokke vahetada.

3.1 Motronic 3.1

Esimesed M-Motronic süsteemid tulid seeriatootmises autodele juba 1979 aastal. M-Motronic sisaldab kõiki Jettronic'u(elektrooniline sissepritse õhulugejaga) funktsioone ning lisaks elektroonilist süütesüsteemi. Selline lahendus aitab põlemisprotsessi paremini kontrollida. Süüdet juhitakse otsingutabelitesse sisestatud väärtuste korrigeerimise teel. Motronic süsteemil on kaks põhilist eesmärki – juhtida mootorisse vastavalt sisenevale õhukogusele õige kogus kütust ja süüdata see kõige optimaalsemal hetkel. Õige kütusekoguse ja süütehetke arvutamiseks on BMW 525 Motronic 3.1 juhtimissüsteemil vajalik mitmete andurite näite:

- väntvõlli asend,
- nukkvõlli asend,
- õhukulu mõõtur,
- drosselklapi asend,
- sisselaskeõhu temperatuur,
- jahutusvedeliku temperatuur,
- lambda andur näit,
- süütevõtme asend.

Täituriteks on :

- pihustid,
- süütepoolid,
- lambda anduri küttekeha relee,
- tühikäigu klapp,
- kütuseaurude eraldussüsteemi klapp,
- kütusepump.

Andurite väärtused on juhtarvutile sisenditeks ning täiturid väljunditeks. Kasutatavate andurite ja täiturite mõistmine aitab analüüsida juhtarvuti tööpõhimõtet.

3.2 Auto mootori juhtarvuti komponendid

Viimase neljakümne aastaga on mootorite juhtimine muutunud üha enam mehaaniliselt kontrollitult elektrooniliselt kontrollitud süsteemidele. Sellega saavutatakse mootorite suurem efektiivsus ning keskkonnasõbralikkus. Samuti on elektrooniliste süsteemidega võimalik mootori tööd jälgida ja teatada võimalikest vigadest. Auto mootori juhtarvutit võib võrrelda gabariitidelt väikese arvutiga, sest ta koosneb samuti nagu tavaline arvuti, järgnevatest komponentidest [8, p. 24;25]:

- keskprotsessor (inglise k *CPU , Central Protcessing Unit*), mis omakorda koosneb kontroll, aritmeetika ja loogika üksusest. Nagu nimedki ütlevad teostavad aritmeetika ning loogika üksused aritmeetilisi ning loogilisi operatsioone ning kontroll üksus teostab instruktsioone, mis on programmi mälust ette antakse. Enamik keskprotsessoreid töötab 8 või 16 bitises süsteemis, kuid on ka 32 bitiseid süsteeme.
- Sisend ja väljund plokid (inglise k *I/O units*). Need käsitlevad andmevahetust väliste seadmete vahel. Välisteks seadmeteks on kõik väljundid ning salvestusseadmed(mälud). Lisaks katkestuste juhtplokk, analoog digitaal teisendid digitaalsed sisendid ja väljundid ning erinevad järjestikühendused(*SPI , CAN*)
- Programmi mälu. Programmi mälus hoitakse kasutaja programmi. Programm määrab kuidas käituvad sisend- või väljundseadmed mingites olukordades. Programm paikneb as *ROM* (inglise k. *Read Only Memory*), *EPROM* (inglise k. *Erasable Programmable Read-only Memory*) või välkmälul(inglise k. Flash memory).
- Andmete mälu ehk *RAM* (inglise k. *Random-access memory*) Selles mälus hoitakse hetkel kasutuses olevaid muutujaid. Tavaliselt on selle mälu suuruseks 2 – 512 Kb.

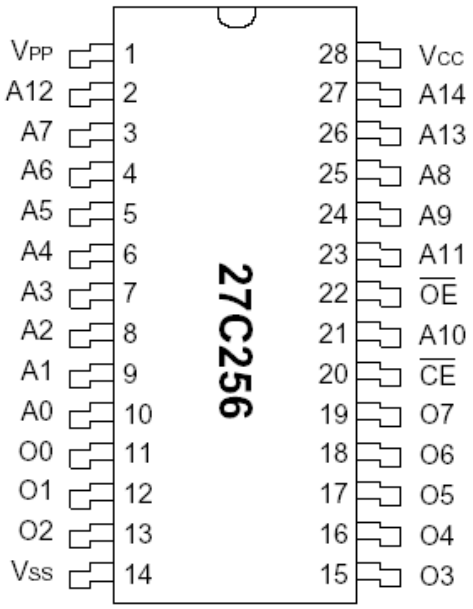
- Siinide süsteem, mis ühendab juhtarvuti erinevaid plokke.
- Taktigeneraator, mis tagab, et kõik protsessoris toimuva oleks sünkroniseeritud ning kindla ajastusega.

Antud töös keskendutakse eelkõige programmi mälule.

Juhtarvuti keskprotsessor arvutab vastavalt sisendparameetritele(näiteks õhukulu) väljundite väärtused(pihustite lahtioleku aeg). Arvutuste teostamiseks vajalikud instruktsioonid ning otsingutabelid asuvad mälus.

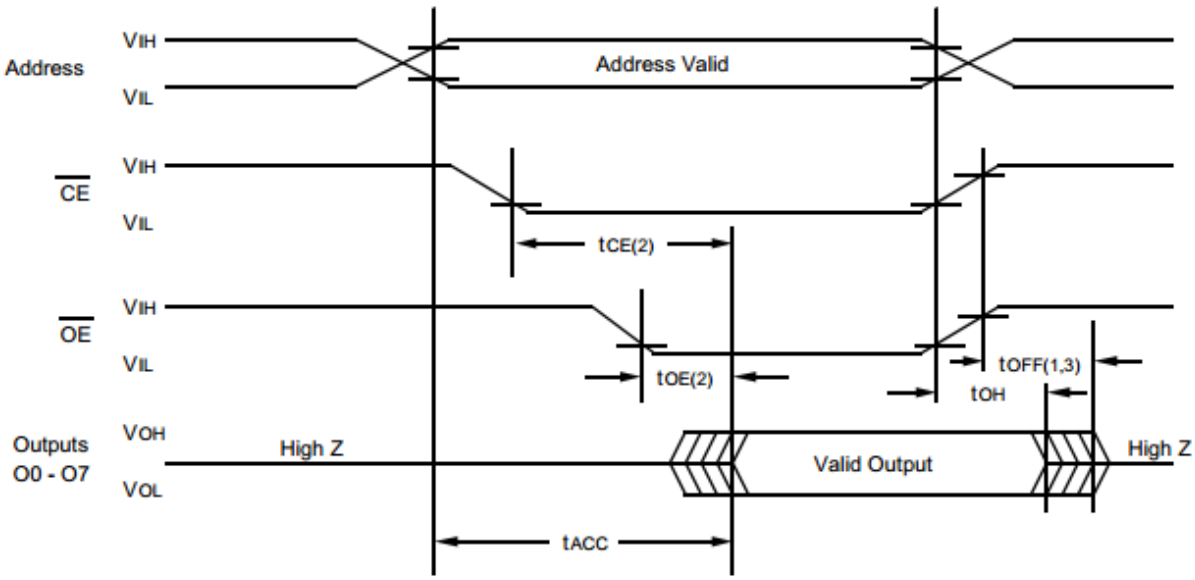
3.3 EPROM 27C256

BMW 525i juhtarvuti osakoodiga 0 261 200 405 kasutab EPROM püsimälu. EPROM on 27C256 tüüpi UV kustutatav ja paigutatud *DIL* (inglise k. *Dual in-line package*) 28 jalaga kesta (Joonis 11). Mälu on trükkplaadile paigutatud läbi pesa nii, et mälu vahetamiseks pole vaja jootmisvahendeid. Mälu suuruseks on 256 kilobitti ja see on organiseeritud 32000 8 bitiseks plokiks. Sellisel mälul on 32767 mälupunkti. 1 bait koosneb 8 bitist (1 bitt on kahendsüsteemis 1 või 0) ja korrutades bittide arvu 32768 saame 262144 bitti ehk 32 Kb kogumälu. 32768 on kuueteistkümnendsüsteemis ehk heksadetsimaalsüsteemis võrdne 8000h (Märkus. Kasutatakse numbri taga tähist „h“ et märkida kuueteistkümnendiksüsteemi). Epromi aadressisüsteemis oleks sellisel juhul mäluvahemik 0000h kuni 7FFFh. [35]



Joonis 11. 27C256 DIL kesta pealtvaade [35]

Sellisel *EPROM*'il on 8 väljund andmeliini (O0-O7) (joonis 12) ning 15 aadressi sisend andmeliini(A0-A14), lisaks V_{CC} positiivse toitepinge jaoks ning V_{SS} negatiivse toitepinge jaoks(maa). V_{PP} on programmeerimis toitepinge, OE ehk *Output Enable* viik ning CE ehk *Chip Enable* viik on sisendite ja väljundite kontrollimiseks. [35]



Joonis 12. 27C256 lugemise ajastus [35]

Selleks, et aktiveerida sisendid, tuleb CE viik viia madalaks signaaliks (Joonis 12) ehk loogiline 0 ehk ligilähedaseks maaga(-0,5 – 0,8V) seejärel loeb mälu aadressiviikide loogilised väärtused. Väljundite aktiveerimiseks tuleb OE viik viia loogiliseks 0 ning seejärel stabiliseeruvad väljund andmeliinid ning võimalik on lugeda 8 bitine väljund. Loogilise 1 või kõrge signaali miinimum väärtus on 2V ning maksimum toitepinge + 1V. Tüüpiline loogiline 1 jääb siiski samaks, mis toitepinge. [35]

Selleks, et keskprotsessor saaks programmimälust mingi väljundi teeb see läbi aadressiviikide päringu vastavasse programmi ossa. Kui kõik aadressiviigid oleks loogiline 1 tehakse päring viimasesse baiti programmis ehk positsioonile 7FFFh, sest 111111111111111 binaaris on võrdne 7FFFh heksadetsimaalsüsteemis. Samuti kui kõik aadressiviigid oleks loogiline 0 tehtaks päring programmiossa 0000h ehk esimesse baiti. Kui aadress oleks näiteks 001000000000000 tehtaks pärig programmiossa 1000h ning kui A12-A14 aadressi viigid oleks loogiline 1 ning ülejäänud loogiline null, tehtaks päring programmiossa 7000h (Tabel 1).

Tabel 1

Päringu asukoha muutumine vastavalt sisenditele

Aadress viik	A 14	A 13	A 12	A 11	A 10	A 9	A 8	A 7	A 6	A 5	A 4	A 3	A 2	A 1	A 0
Binaarväärtus	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Heksadetsimaalsüsteemi väärtus	1			0				0				0			
Binaarväärtus	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Heksadetsimaalsüsteemi väärtus	7			0				0				0			

Originaal juhtarvutis olev mälukiip on UV kustutatav. Mikrokiibi korpuse peal on kvarts klaasist aken, mis võimaldab ultraviolettkiirgusel pääseda kiibi sees olevate transistoriteni ning muuta igas mälupunktis olev informatsioon loogiliseks 1. Selline UV valgusega kustutamismeetod võtab tavaliselt aega 15-20 minutit spetsiaalse lambi all ja tavalise päikesevalguse käes isegi mitu nädalat. Pärast elektrilist programmeerimist muudetakse vastavates mälupunktides informatsioon loogiliseks 0. [36, pp. 9-4]

Nagu näha, on sellises mälukiibis informatsiooni muutmine suhteliselt keeruline ja aeganõudev protsess. Õnneks on tänapäeval saadaval mugavamaid lahendusi. Nimelt EEPROM(Erasable Programmable Read-only Memory) mälude näol. Need mälud võimaldavad kustutamist ja programmeerimist teostada elektriliselt. Kõrgemad pinged genereeritakse kustutamiseks mikrokiibi sees. [36, pp. 9-9]

Antud töös kasutatakse mälude lugemiseks ning programmeerimiseks MiniPro TL866CS programmaatorit ja sellega kaasa tulnud MiniPro Programmer tarkvara versiooniga 6.0

3.4 27SF512

Selleks, et juhtarvuti programmi mugavalt muuta, tuleb asendada kasutuses olev 27C seeria UV kustutatav EPROM mälul elektriliselt kustutatava 27SF seeria mälu vastu. Saadaval on nii 256Kbit, 512Kbit, 1000Kbit kui ka 2000 Kbit versioone [37]. Antud rakendusse on parim variant 512Kbitine mikrokiip, kuna seda on saada täpselt samasuguses DIL28 korpuses nagu 27C256 varianti. 1 ja 2Kbitised versioonid on saadaval vaid 32 viiguga korpuses). 512Kbitine mälu mahutab kaks korda

rohkem informatsiooni kui originaalne 27C256. 27SF512 mälu on parameetrite poolest väga sarnane 27C256 . Tähtsamad parameetrid on välja toodud järgnevas tabelis:

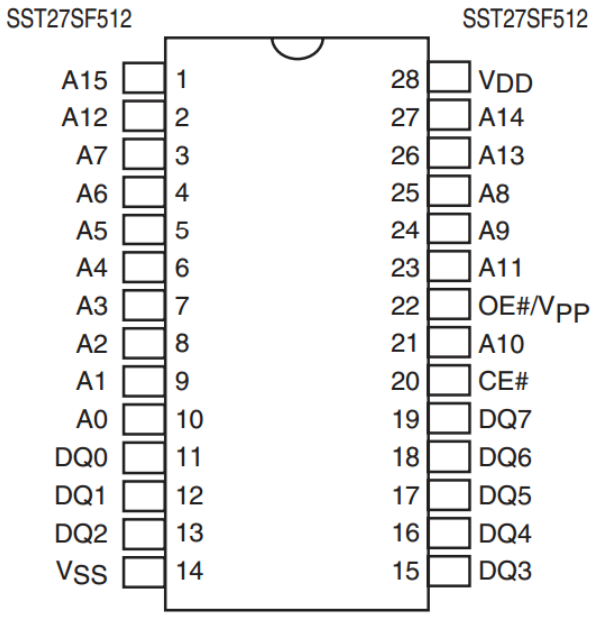
Tabel 2

Tähtsamate parameetrite võrdlus erinevatel mäludel [35] [37]

	27C256	27SF512
Toitepinge (V)	4,5-5,5	4,5-5,5
Ühe lugemistsüklile kuluv aeg(ns)	70-90	70
Voolutarve lugemishetkel(mA)	20-25	30
Madala signaal(V)	0-0,45	0-0,2
Kõrge signaal(V)	2,4- V_{CC}	2,4-V_{CC}

Nagu näha, siis 27SF512 on ideaalne asendus originaalile. 10mA suurem voolutarve ei oma terviksüsteemis mingit tähtsust. Kõik ülejäänud parameetrid on suurema mahuga mälul samad või paremad kui UV kustutaval kiibil.

Mõlemad mikrokiibid on *DIL*28 korpuses, kuid 1. viigi funktsioon on teine (Joonis 11)(Joonis 13)



Joonis 13. 27SF512 *DIL* kesta pealtvaade [37]

Kuna 27SF512 kiibil on kaks korda rohkem mälu, siis on viik järjekorranumbriga 1 kasutusel kui sisendviik kõrgeimasse aadressiplokki päringute tegemiseks ning viik 22 toimib nii *Output Ennable* kui ka programmeerimispinge viigina. Kuna lugemise ajal $V_{PP} = V_{CC}$ [35], siis tähendaks see

27SF512 mälu jaoks kõrget signaali ning olenemata A0-A14 viikide seisundist, algaks programmi lugemine alates 8000h ning kui kõik 16 aadressiviiki oleks binaarväärtuses 1, siis loetaks aadressi FFFFh. Järelikult kui kirjutada 32Kb programm alates aadressist 8000h, siis toimib 27SF512 kiip täpselt samamoodi nagu 27C256.

3.5 Mäluplokkide vahetamine

Eelnevalt kirjeldatud asenduse korral on võimalik mälukiipi lihtsasti ümber programmeerida, kuid kasutamata jääks pool mälumahust. Sinna on võimalk juhtprogramm dubleerida, kuid kasutades teisi väärtusi näiteks süüte või kütuse otsingutabelites. Samuti on võimalik tuua ühe juhtprogrammi pööretepiiraja madalamale ja teisel kõrgemale, et simuleerida näiteks kohaltmineku abi (*launch control*). Seega juhtarvuti keskprotsessor teeks päringu mälupunkti 70FBh, mis on pihusti lahtioleku aeg 100% koormusega 6400p/min juures, siis aadresside viikide väärtused oleksid järgmised:

Tabel 3

Sisendite väärtused 70FBh mälupunkti

Aadress viik	A 14	A 13	A 12	A 11	A 10	A 9	A 8	A 7	A 6	A 5	A 4	A 3	A 2	A 1	A 0
Binaarväärtus	1	1	1	0	0	0	0	1	1	1	1	1	0	1	1
Heksadetsimaalsüsteemi väärtus	7			0				F				B			

Lisades 16 aadressi viigi ja viies ta väärtuse loogiliseks 0, saadakse täpselt samasugune kuuetestkümnendiksüsteemi väärtuse mis ülal tabelis(Tabel 3) Kui 16. viigi väärtus oleks 1, liigub program 8000h mälupunkti võrra kõrgemale (Tabel 4).

Tabel 4

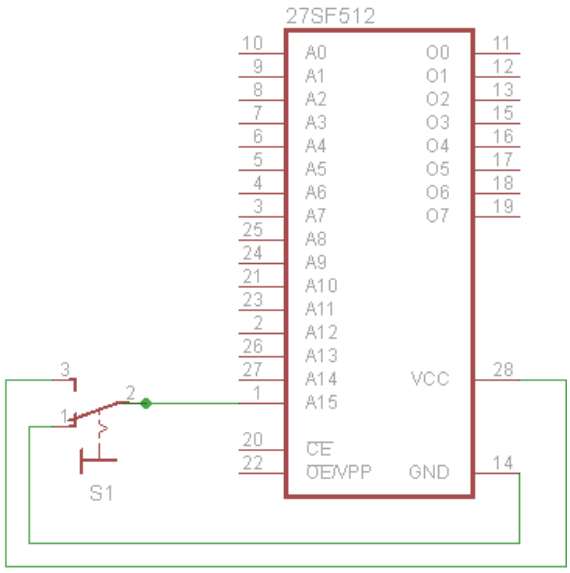
Sisendite väärtused F0FB mälupunkti

Aadress viik	A 15	A 14	A 13	A 12	A 11	A 10	A 9	A 8	A 7	A 6	A 5	A 4	A 3	A 2	A 1	A 0
Binaarväärtus	1	1	1	1	0	0	0	0	1	1	1	1	1	0	1	1
Heksadetsimaalsüsteemi väärtus	F				0				F				B			

Mälupunkti kaugus programmi alguspunktist jääb täpselt samaks. Kui kaks juhtprogrammi on paigutatud mälukiibile üksteise järgi nii, et esimene hakkab aadressilt 0000h ja lõppeb 7FFFh ning teine hakkab 8000h ja lõppeb FFFFh, saab neid vahetada muutes A15 aadressiviigi loogilist väärtust. Kui väärtuseks on 0 või madal signaal, kasutatakse mälu esimest poolt ning kui väärtuseks on 1 või kõrge signaal, kasutatakse mälu teist poolt.

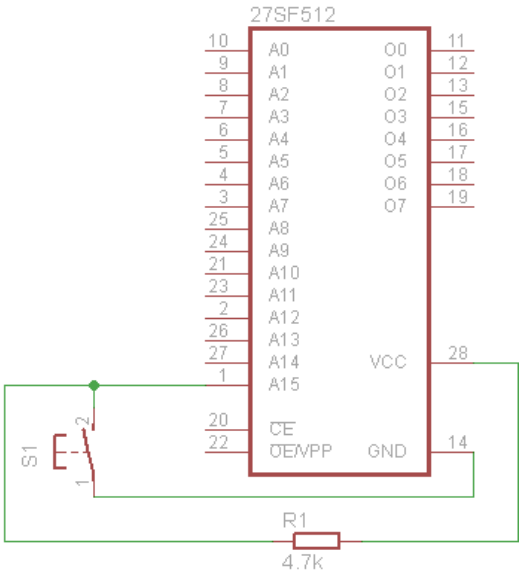
3.6 Programmide vahetamine mittetöötava mootoriga

Selleks, et saaks juhtida 27SF512 mälukiibi A15 viigi väärtust tuleb viia kõrge signaali saamiseks potentsiaal võrdseks V_{CC} pingega ning madala signaali korral V_{SS} pingega. Seda võiks teha tavalise kahepositsioonilise lülitiga(Joonis 14).



Joonis 14. Lülitiga mäluplokkide vahetamine

Teine ja parem variant oleks kasutada tõmbe takistit(*pull-up*), et viigi väärtust vahetada. Sellisel juhul tuleks juhtarvutist välja vedada vaid kaks juhet ning samuti ei oleks need juhtmed ühendatud mikrokiibi toitepingega, sest juhtme purunemise korral võib see mälu ja ka kogu juhtarvutit kahjustada(auto kere on ühendatud massiga) (Joonis15). Samuti vähendab takisti kasutamine skeemis häireid kuna nii puuduks pikk toitepingega juhe. Lüliti 14. viiki mineva otsa võib ühendada ka auto kere massi, sest mikrokiibi maa ja auto maa on üks ja sama potentsiaal (Märkus. Sisendviigi väärtus võib -0,5-0,8V maast erineda. Erinevus võib tekkida pingelangudest süsteemis).



Joonis 15. Lülitiga mäluplokkide vahetamine kasutades tõmbetakistit

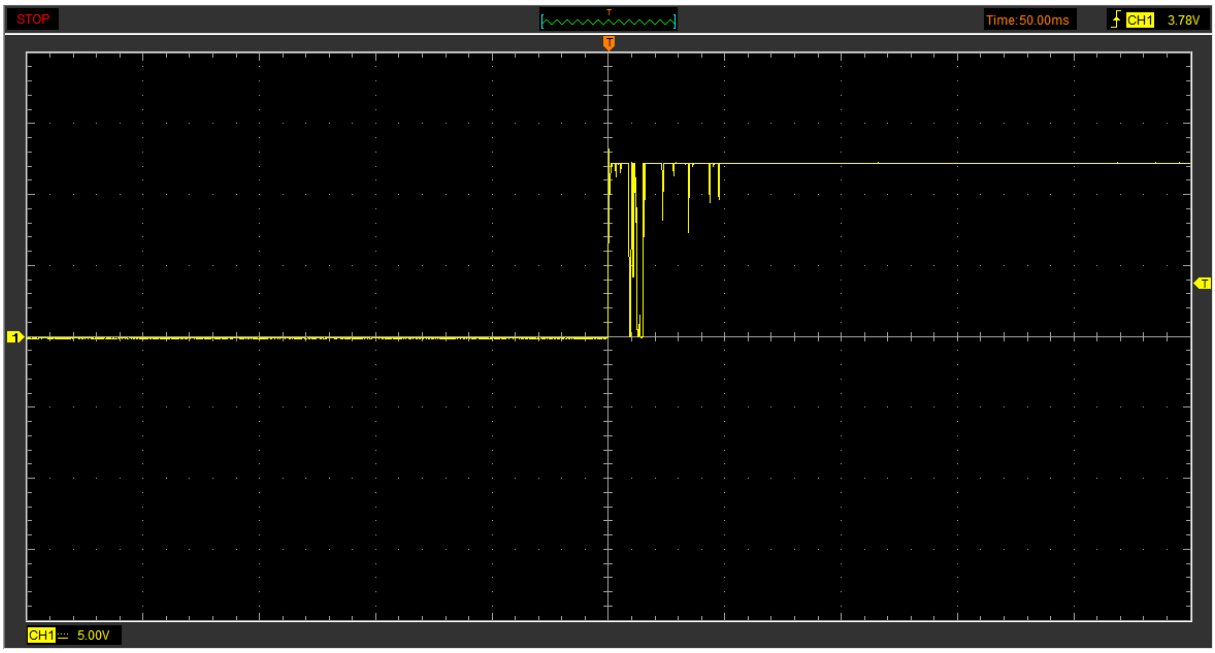
Takisti väärtus ei ole väga tähtis kuid võiks jääda vahemikku 4,7KΩ-10KΩ oomi kuna mikrokiibi sisendtakistus on väga suur.

Selline lahendus töötaks väga hästi, kui lülitit lülitada olukorras kus auto süüde on välja keeratud, kuid mitte mootori töötamise ajal.

Programmi peab saama muuta ka mootori töötamise ajal, et kasutada näiteks eelpool mainitud kohaltmineku abi. Samuti kui autot peaks kasutama võõras inimene, kes ei oma informatsiooni nupu funktsiooni kohta(vajutatakse kogemata), siis peab olema tagatud, et mootor ei kahjustuks. Samuti võib antud lahendust kasutada mõnel muul autol, mille juhtimissüsteem kasutab paralleelmälu ja erinevaid seadeid on vaja näiteks külma mootori puhul ja sooja mootori puhul. Viimane võib kasuks olla ülelaaditud mootorite korral. Ka seeriatootmises omavad autod lubavad väljundkarrakteristikat muuta sõidu ja mootori töötamise ajal [2].

3.7 Lüliti müra

Tavalised sisse-välja lülitid tekitavad igal lülitamisel müra, mis võib kesta ligi 0,5mS (Joonis 16)



Joonis 16. Lüliti müra ostsilloskoobi ekraanil

Märtkused.
1. Mõõtmine on teostatud pingel 12V.
2. Koormuseks on vaid ostsilloskoobi sisendtakistus.
3. Lülitiks kasutati m2013ss1w01 lülit [48].

Mälu täiskiirusel töötamise juures, kus ühe mälupunkti lugemine võtab 70ns on 0,5ms pikkune aeg väga pikk aeg ning programm võib käia kõrgemast mäluplokist madalamasse mitukümmend korda enne kui lüliti lõplik väärtus on saavutatud.

Veel halvem juhtum on see, kui aadressiviigi väärtust muudetakse täpselt väljundi lugemise hetkel(Joonis 12). Sellisel juhul võivad väljundi viigid muuta oma väärtusi juhuslikult ning kui päring tehakse näiteks mälupunkti, kus peaks olema süütenurk enne ülemist surnud seisu 22°, aga rikutud väljundi puhul loetakse see 55°, siis saab kahjustada ka mootor.

3.8 Programmi vahetamine mootori töötamise ajal

Selleks, et vältida lüliti poolt tekitatud müra ning tagada, et lülitus toimub hetkel kui mälu ei loeta, tuleb konstrueerida ümberlülitusskeem, mis arvestab nende tingimustega.

Mälu ei loeta hetkel, kui CE ja OE viigid on loogilised 1(Joonis 12) järelikult tuleks kasutada loogilist operatsiooni NING (Tabel 5) [3, p. 27].

Tabel 5.

Ning lülituse tõeväärtustabel [3, p. 27]

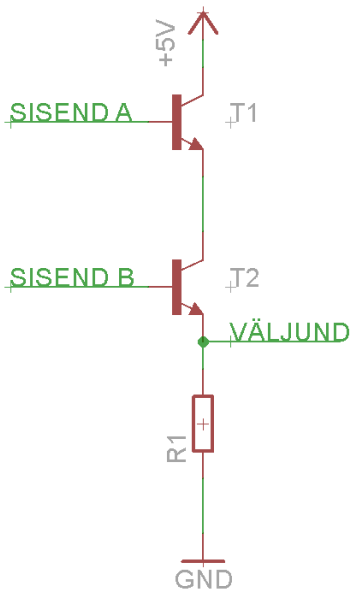
Sisend		Väljund
A	B	A ning B
0	0	0
0	1	0
1	0	0
1	1	1

Kui lülitus on toimunud, peab väärtus jääma valitud tasemele nii kauaks, kuni süüde välja keeratakse või valitakse teine mäluplokk. Selle tagab d-tüüpi triger(inglise k *flip-flop*).

Lüliti müra vältimiseks koostatakse kahes transistorist koosnev skeem, mis muudab olekut esimese pingepiigi saabudes.

3.9 „Ning“ lülitus

Lülitus koosneb kahest NPN tüüpi transiostorist, mille baasid on ühendatud CE ja OE viikidega mälukiibil. Esimese transistori kollektor on ühendatud toitepinge ning teise kollektor esimese emittriga. Teise transistori emitter on läbi *pull-down* takisti ühendatud maaga. Selline konfiguratsioon toimib kui „ning“ lülitus sest väljund on kõrge vaid siis kui mõlemad transistorid on avatud. Transistorite sulgudes tagab *pull-down* takisti, et pinge ühtlustuks maaga(Joonis 17)(Lisa 2)

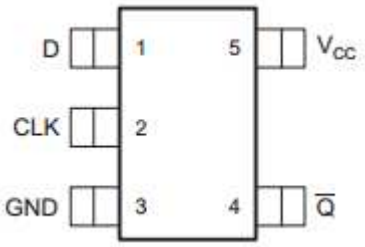


Joonis 17. NING lülitus kasutades kahte transistorit

3.10 D-tüüpi triger

Valiti SN74LVC1G80 triger, kuna selle parameetrid on antud lahendusse ideaalsed, seda valmistatakse pindmontaažile mõeldud korpuses (projekteeritav ümberlülitusskeem hakkab paiknema juhtarvuti korpuses sees, kus ruum on piiratud) ja kohalikust kaubandusvõrgus on mikroskeem olemas.

SN74LVC1G80 sisaldab endas mitmeid loogikalülitusi ning vaheldeid, kuid omab kahte sisendit ja ühte väljundit. Lisaks on viigid toitepinge ja maa jaoks(Joonis 18).



Joonis 18. SN74LVC1G80 D-tüüpi trigeri pealtvaade [38]

Kasutatava trigeri tööpõhimõtet iseloomustab järgmine tabel (Tabel 6).

Tabel 6.

SN74LVC1G80 väljundite muutus vastavalt sisenditele [38]

Sisendid		Väljund $\overline{Q}$
CLK	D	
↑ kasvav	1	0
↑ kasvav	0	1
0	X	Viimatine $\overline{Q}$

Nagu näha, siis väljund on sisendi suhtes inverteeritud, kuid kasutatavas rakenduses ei oma see tähtsust(lüliti saab paigutada mõlemat pidi). Samuti väärib märkimist, et väljundit muutuvad CLK signaali kasvamise hetkel, ehk triger on aktiveeritud tõusval signaaliserval.

CLK viik on taktisignaali jaos. Taktisignaal antud juhul peab olema ühendatud „ning“ lülituse väljundiga. Sellisel juhul muutub SN74LVC1G80 väljund juhul kui „ning“ lülituse väärtus on tõene(Lisa 3). [38]

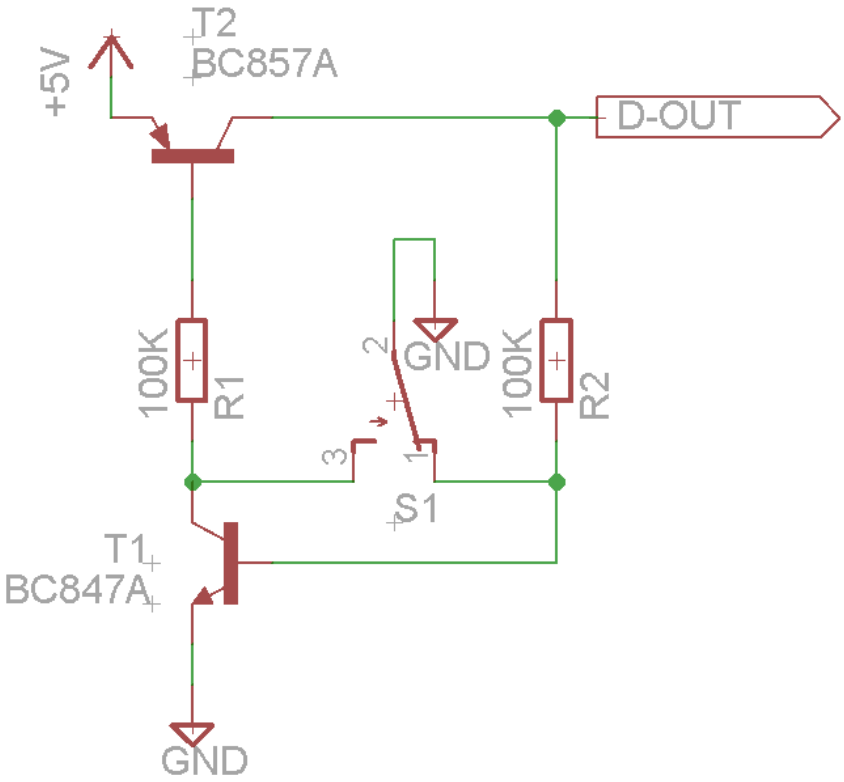
D viik on andmete jaoks ja saab olla kas kõrge või madal. Kui D viik on kõrge ja CLK sisend on kasvav muudetakse väljund madalaks ning vastupidi(Tabel 6)

$\overline{Q}$ on väljund ja muutub vastavalt eelnevalt kirjeldatud sisenditele. Kõrge signaal on samal tasemel VCC pingega ning madala signaali pinge on maa

Toitepinge peab jääma vahemikku -0,5 – 6,5V [38]. Kuna SST27SF512 mälukiip töötab pingega 5V, siis on see sobiv.

3.11 Lüliti müra vältimine

Lüliti müra saab vältida koostades skeemi, mis lülitub sisse ja välja esimese pingepiigi saabudes. Seda on võimalik lahendada kõigest kahe transistoriga. Üks nendest on PNP tüüpi ja teine NPN tüüpi(Joonis19).

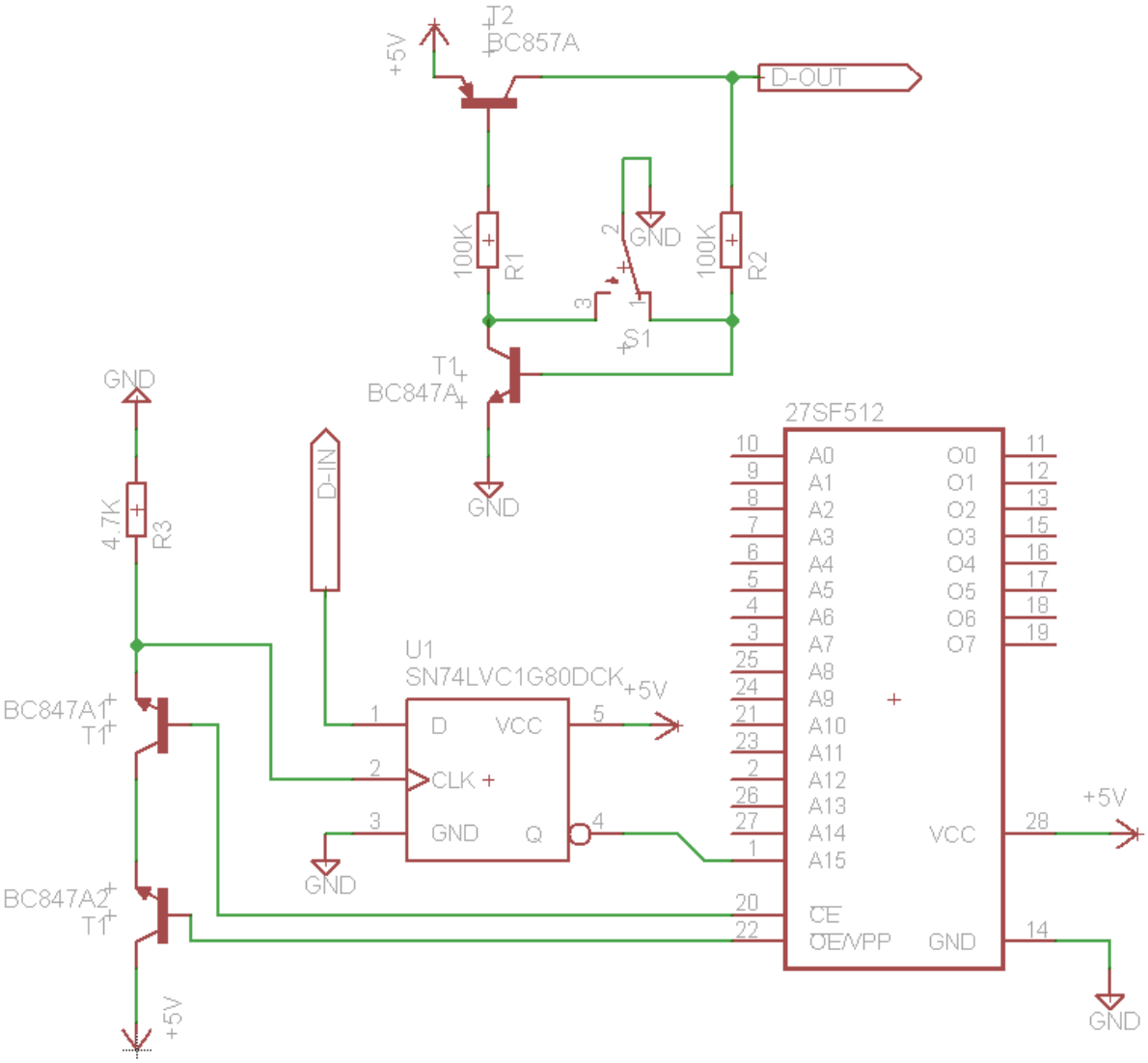


Joonis 19. Lüliti müra vältimise skeem

Lühistades PNP transistori T2 baasi läbi 100 kΩ takisti maasse on see transistor täielikult avanenud ning vool saab liikuda kollektorist emitterisse. Samal hetkel avaneb ka NPN transistor T1, mis tagab, et olenemata kas lüliti on täielikult ümber lülitunud või mitte, saab vool liikuda maasse ja transistor T2 püsib avatud. Samuti muutub väljund D_{OUT} ligilähedaseks toitepingega, kuid jääb alati sellest madalamale (PN siirde pingelangu jagu). See aga ei oma tähtsust, sest pinge peab jääma kõrgemale, kui SN74LVC1G80 sisendi loogilise 1 pinge alampiir, milleks on 5 V toitepinge korral 0,7*5=3,5(V). Lüliti ümber lülitades jääb lüliti kontakt 3 avatuks ning lühistatakse NPN transistori T1 baas maaga, seega T2 transistori baasist ei saa vool enam maasse liikuda ning see sulgub. Väljundisse D_{OUT} on siis pinge 0V kuna see on läbi 100 kΩ takisti ühendatud maaga. Lüliti juhtmed on ühendatud maaga, seega pole ohtu, et juhtme purunedes kahjustuksid ülejäänud komponendid – väljund on siis alati madal signaal. Takisti väärtused on valitud suured, et vältida skeemi liigset voolutarbimist. Transistorid on valitud samast seeriast ning on sarnaste parameetritega [39] [40].

3.12 Kasutatav programmide ümberlülitusskeem

Kombineerides kolm eraldiseisvat plokki ühiseks skeemiks, on võimalik teostada ohutu lülitus ühelt programmiosalt teise(Joonis 20).



Joonis 20. Programmide ümberlülitusskeem

Simulatsioon näitas samuti, et skeem toimib nagu eeldatud

Sama loogikat kasutades oleks võimalik kasutada ka suurema mahtuvusega mälukiipi ning näiteks nelja erineva seadega programmi. Valida tuleks mitme kanaliga trigerit näiteks 74HC175 seeriast [41]. Lüliti ja müravastase skeemi võiks sellisel juhul asendada mikrokontrolleriga, mis erinevaid mäluplokke valib.

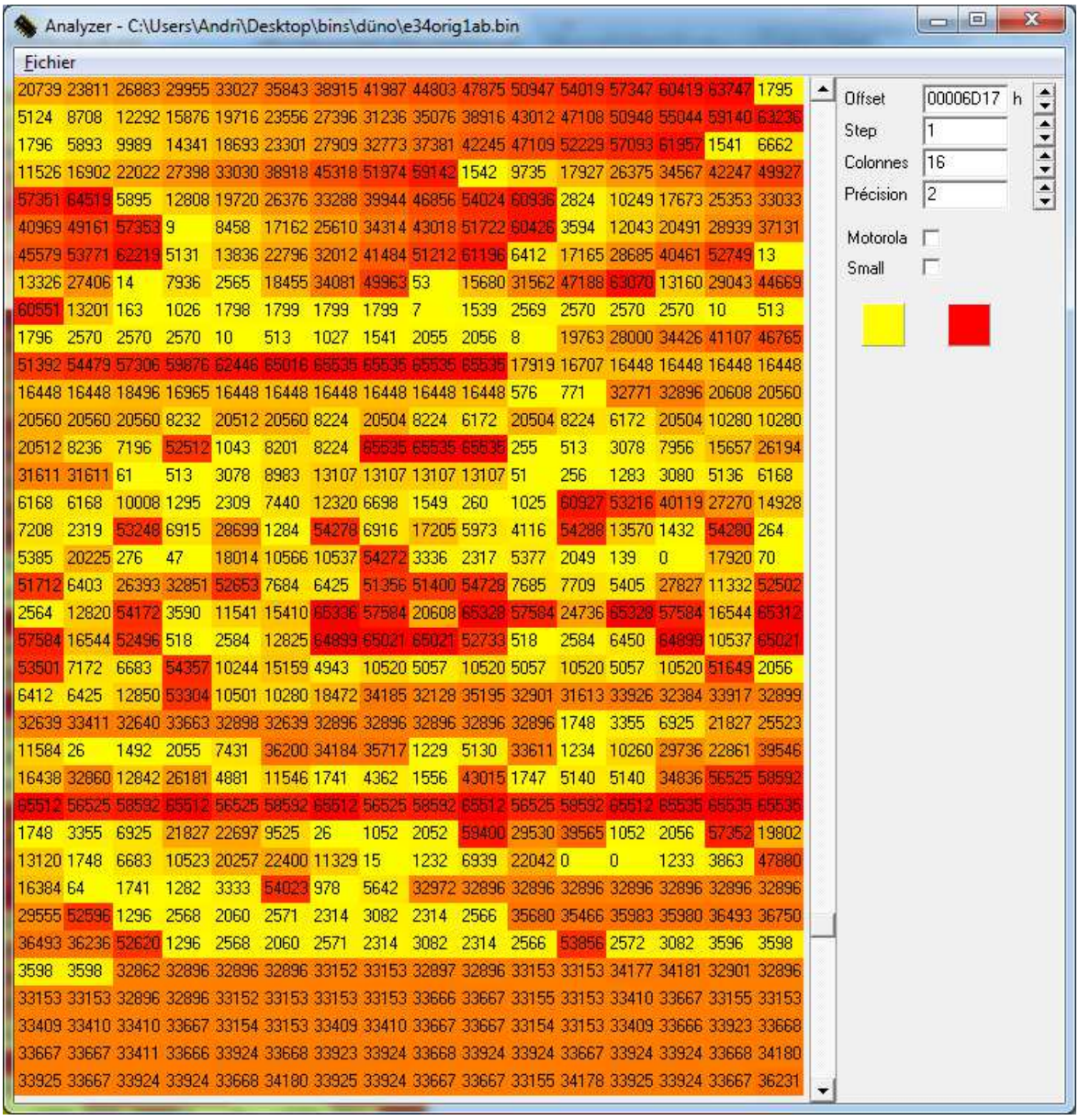
4. OTSINGUTABELID

Selles peatükis kirjeldatakse kuidas lokaliseerida, mõista ja visualiseerida programmimälus paiknevaid otsingutabeleid. Lisaks kontrollitakse info paikapidavust veojõustendis väljundkarakteristikat mõõtes.

4.1 Otsingutabelite leidmine

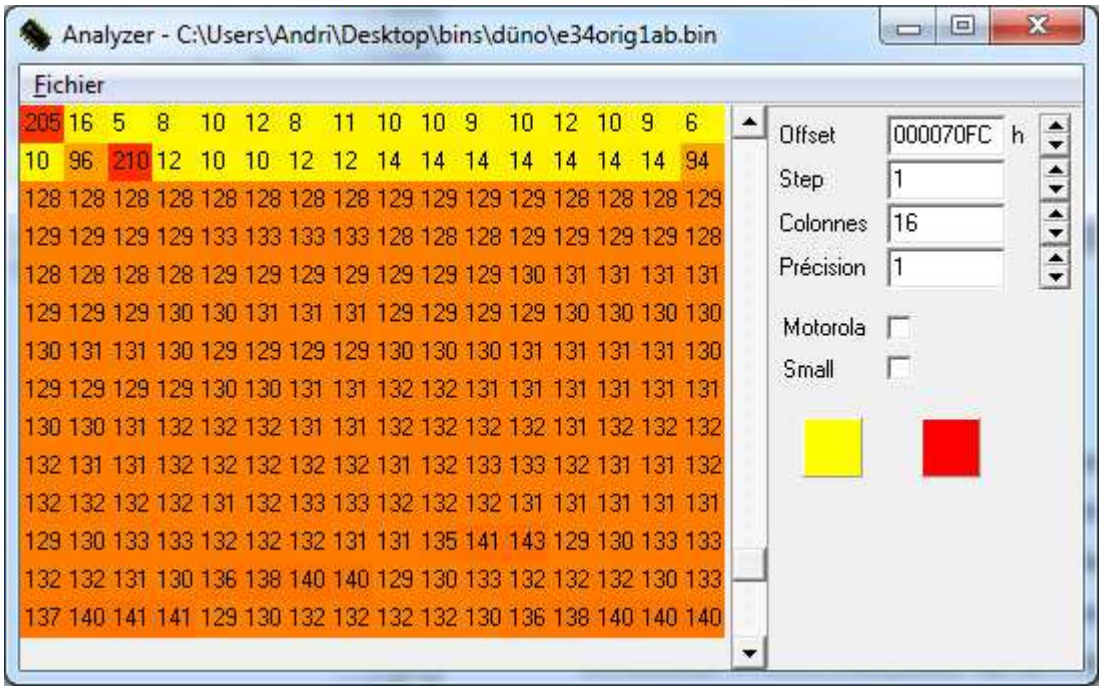
Juhtarvuti töö optimeerimiseks ning kiirendamiseks on paljud muutujad, mis juhtarvuti tööks vajalikud on, paigutatud otsingutabelitesse. Nii välditakse korduvaid arvutusi, mis aeglustavad protsessori tööd. Sellistesse kaartidesse on paigutatud näiteks baasväärtused pihusti lahtiolekuajaks ning süütehetkeks. Järgnevalt on kirjeldatud ühte võimalust kaartide leidmiseks.

Programmimälust kaartide leidmiseks kasutatakse tasuta saadaval olevat programmi „Analyzer“, mis teisendab juhtarvuti programmimälust loetud binaarformaadis 16 bitise koodi detsimaalsüsteemi ning värvib väärtused gradientselt suurenedes valitud toonidesse. Selliselt on võimalik visualiseerida kasvavate väärtustega kaarte ja leida nende asukohad.(Joonis 21)



Joonis 21. Ekraanitõmmis „Analyzer" programmist

Ekraanitõmmise ülaosas on näha järjest suurenevaid väärtusi ning allosas palju sarnaseid väärtusi. Ülejäänud kaootiliselt muutuvatest väärtustest on need programmiosad selgesti eraldatavad. Võib oletada, et tegemist on otsingutabelitega. Samuti on enne sarnaseid väärtus märgata kuus väiksemat väärtust seejärel üks suur, siis kaheksa väiksemat väärtust ning üks suurem väärtus. Väärtused on teisendatud 16 bitist, sest suuri otsingutabeleid on nii kergem märgata, aga juhtarvuti on 8 bitine seega teisendatakse väärtused samuti kaheksabitilisteks(Joonis 22).



Joonis 22. Koodiosa 8 bitiste väärtustega

Nüüd on näha, et sarnastele väärtustele(128-143) eelneb 13 väiksemat väärtust(12-94), suurem väärtus(210) ning sama süsteemiga 17 väikest väärtust(16 -96) ning jällegi suurem väärtus. Sama otsingutabel heksadetsimaalsüsteemis väärtustega näeb välja järgmiselt:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000070F0	8C	8C	8C	8D	8E	8E	8F	8D	8E	8C	8D	8C	CD	10	05	08
00007100	0A	0C	08	0B	0A	0A	09	0A	0C	0A	09	06	0A	60	D2	0C
00007110	0A	0A	0C	0C	0E	0E	0E	0E	0E	0E	0E	5E	80	80	80	80
00007120	80	80	80	80	81	81	81	81	80	80	80	81	81	81	81	81
00007130	85	85	85	85	80	80	80	81	81	81	81	80	80	80	80	80
00007140	81	81	81	81	81	81	81	82	83	83	83	83	81	81	81	82
00007150	82	83	83	83	81	81	81	81	82	82	82	82	82	83	83	82
00007160	81	81	81	81	82	82	82	83	83	83	83	82	81	81	81	81
00007170	82	82	83	83	84	84	83	83	83	83	83	83	82	82	83	84
00007180	84	84	83	83	84	84	84	84	83	84	84	84	84	83	83	84
00007190	84	84	84	84	83	84	85	85	84	83	83	84	84	84	84	84
000071A0	83	84	85	85	84	84	84	83	83	83	83	83	81	82	85	85
000071B0	84	84	84	83	83	87	8D	8F	81	82	85	85	84	84	83	82
000071C0	88	8A	8C	8C	81	82	85	84	84	84	82	85	89	8C	8D	8D
000071D0	81	82	84	84	84	84	82	88	8A	8C	8C	8C	D4	04	1B	28

Joonis 23. Otsingutabel heksadetsimaalsüsteemis positsioonil 70FCh

4.2 Otsingutabelite vorming

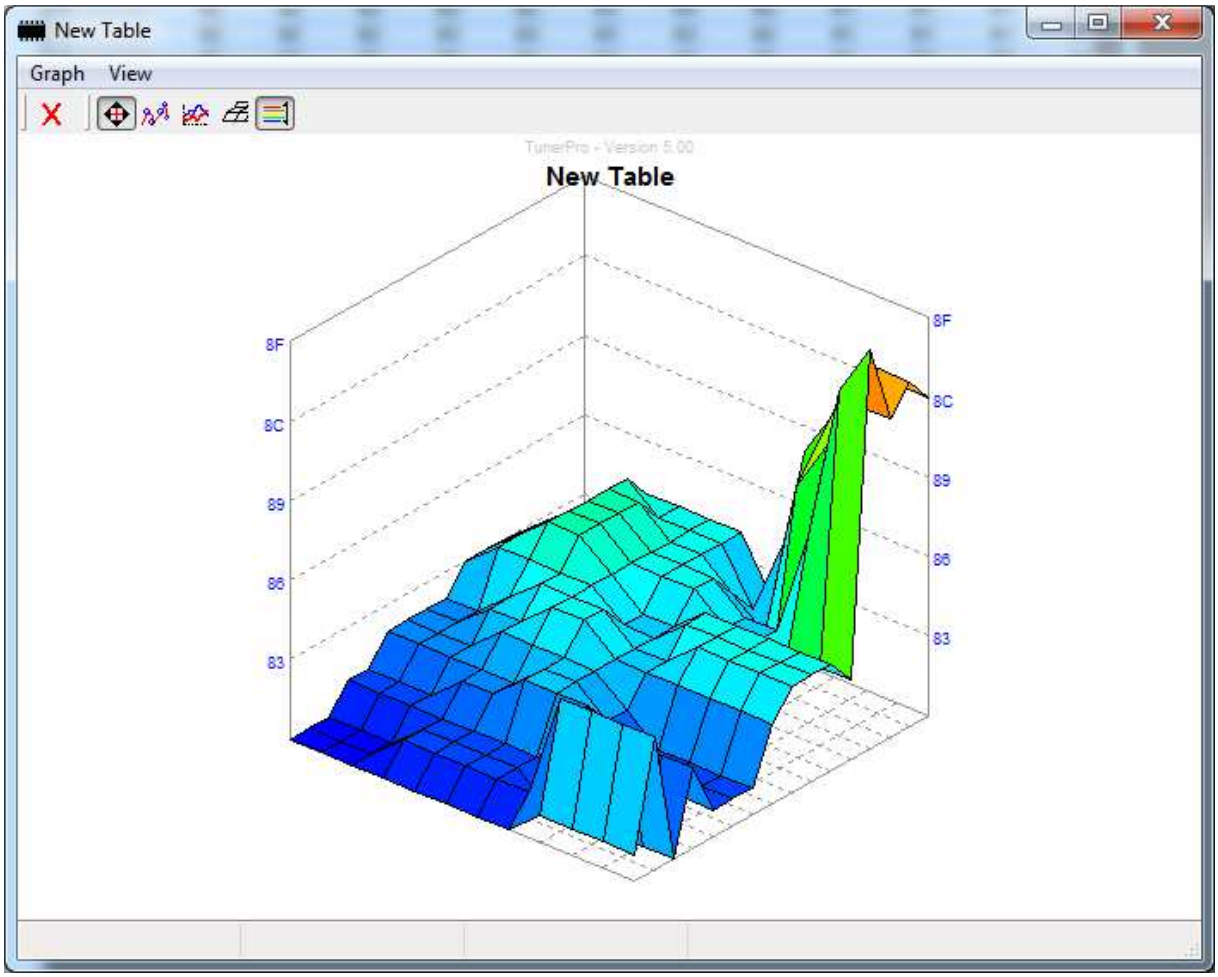
Alates Motronic ML3.1 süsteemidest kuni OBD2 süsteemideni kasutati ühesugust otsingutabelivormingut [42]. Kõikidel kaartidel on enne otsingutabelite väärtusi märgitud teljed, nende pikkus ning telgede väärtused (Joonis23). Joonisel on vormingu mõistmiseks erinevad tähendused tähistatud erinevate värvidega, kus:

CDh(punane)	- X telje tähistus;
10h(kollane)	- kuuetesitkümnendsüsteemis X telje pikkus baitides;
70FEh – 710Dh (roheline)(aadressi väärtused)	- X telje väärtused;
D2h(punane)	- Y telje tähistus;
0Ch(kollane)	- kuuetesitkümnendsüsteemis Y telje pikkus baitides;
7110h-711Bh(roheline)(aadressi väärtused)	- Y telje väärtused;
711Ch-71DBh	- otsingutabelis paiknevad andmed.

Andmed jagunevad telgede vahel nii, et esimesena on tabelis X telje pikkuse jagu andmeid seejärel Y telje pikkuse jagu andmeid, siis jälle X jne. Seega esimesed 16(10h=16 detsimaalsüsteemis) baiti kuuluvad X teljele ning järgmised 12(0Ch = 12 detsimaalsüsteemis) baiti Y teljele. [42]

4.3 Otsingtabelite visualiseerimine ja funktsiooni selgitamine

Teades, kuidas otsingutabelid moodustuvad, on võimalik need andmete paremaks mõistmiseks visualiseerida. Antud töös kasutatakse selleks Tuner Pro RT 5.0 tarkvara [43], mis võimaldab defineerida kaartide asukohad ja teljed ning visualiseerida nii kahe- kui ka kolmedimensioonilisi kaarte. Programm on tasuta saadaval, kuid kasutamiseks on soovitav teha annetus autorile või oodata 10 sekundit peale programmi avanemist.



Joonis 24. Eelpool leitud otsingutabel visualiseerituna 3D formaadis

Arvestades otsingutabeli suurust(16*12) ning väärtuste kasvavat tendentsi võib oletada, et tegu on kütusekoguse(pihusti lahtiolekuaeg) otsingutabeliga. Järelikult peab X teljeks olema mootoripöörded ning Y teljeks koormus. [44, p. 154]

Et telg oleks loetav, tuleb detsimaalsüsteemis väärtused korrutada 40, sest 8 bitises(2^8) süsteemis on maksimaalselt 255 väärtust(nulli ei arvesta kuna null tähendaks mootori seiskumist) ja motronic juhtarvutid projekteeriti töötama kuni10200p/min [45]. Seega ühe hamba väärtus on 40 (Tabel 7).

Tabel 7

X-telje väärtused

Väärtus heksadetsimaal-süsteemis	5	8	0A	0C	8	0B	0A	0A	9	0A	0C	0A	9	6	0A	60
Väärtus detismaal-süsteemis	5	8	10	12	8	11	10	10	9	10	12	10	9	6	10	96
Pöörded (*100 p/min)	5,8	7,8	11	15	19,8	23	27,4	31,4	35,4	39	43	47,8	51,8	55,4	57,8	61,8

Y telje väärtused peavad jääma vahemikku 0-100 %. Kasutades sama loogikat saame paika panna koormussignaali telje(Tabel 8). Konstandina kasutatakse sellisel juhul 7,04(sellise konstandiga saame vahemiku 0-100%).

Tabel 8

Y telje väärtused

Väärtus heksadetsimaal-süsteemis	0A	0A	0C	0C	0E	0E	0E	0E	0E	0E	0E	5E
Väärtus detismaal-süsteemis	10	10	12	12	14	14	14	14	14	14	14	94
Koormus %	0	7,04	14,08	22,53	30,98	40,83	50,69	60,54	70,40	80,27	90,11	99,99

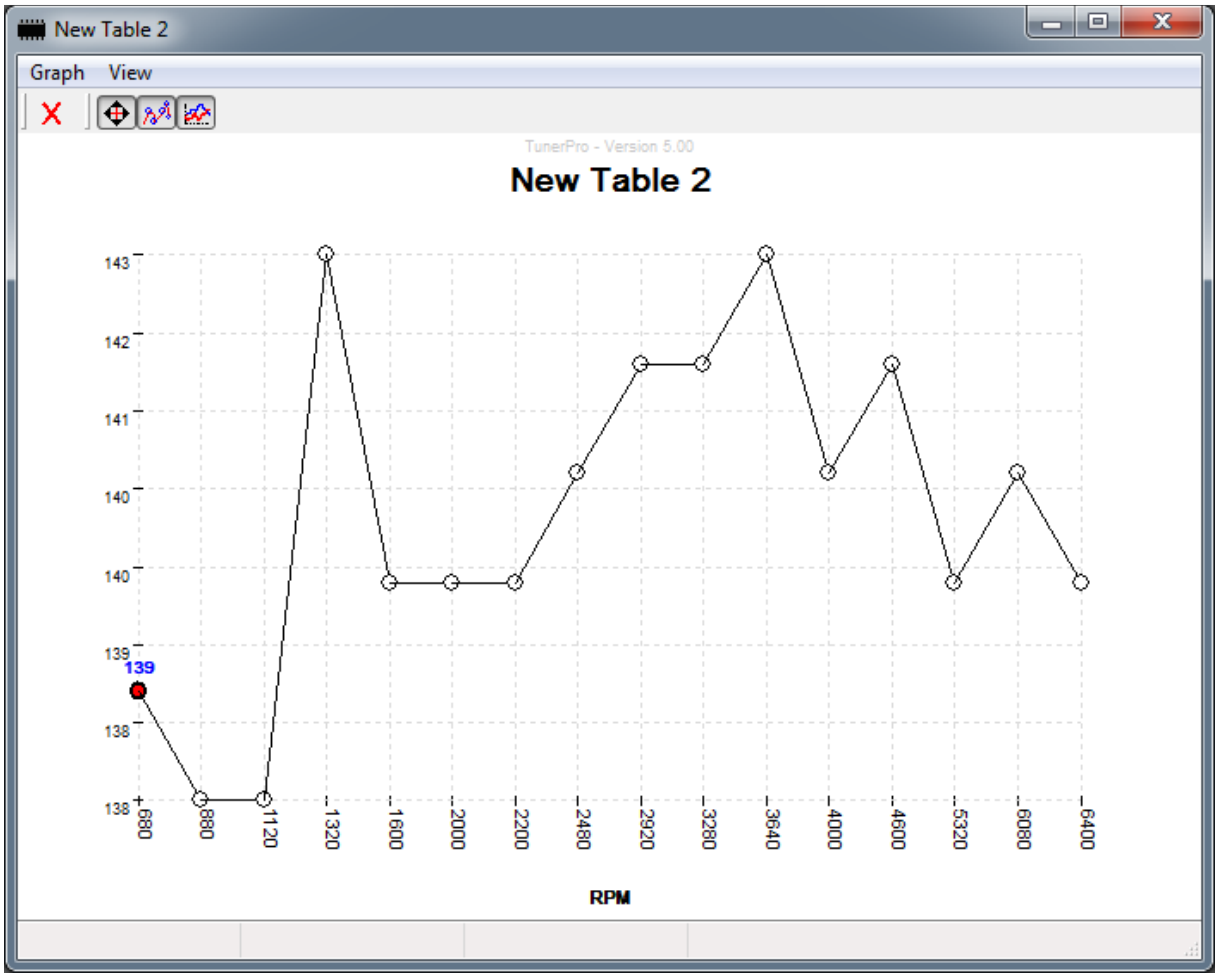
Reaalselt otsingutabeleid muutes ei pea olema telgede väärtused absoluutselt täpsed, kuna kahe otsingutabeli punkti vahele jäävad väärtused interpoleeritakse läheduses olevatest väärtustest. Seega, kui teljed on nihkes, ei mõjuta see lõpptulemust. Samuti on mootori töös palju muutujaid ning tabelite resolutsioon on suhteliselt väike, et kogu pööretevahemikus parameetrid ideaalseks saada. [44, p. 278]

Eelnevast on näha, et CDh tähendab mootoripöördeid. Otsides programmist samasuguseid X telje väärtusi on leitav veel 1 kaart, sarnaste väärtustega eelnevale (Joonis 25).

```
000070D0  80 80 80 80 80 80 80 73 73 74 CD 10 05 08 0A 0C
000070E0  08 0B 0A 0A 09 0A 0C 0A 09 06 0A 60 8B 8A 8A 8F
000070F0  8C 8C 8C 8D 8E 8E 8F 8D 8E 8C 8D 8C CD 10 05 08
```

Joonis 25. Otsingutabel positsioonil 70DB

Teisendades väärtused kümnendsüsteemi, visualiseerides väärtused (Joonis 26) ja arvestades kaardi suurust võib väita, et tegemist on 100% koormusega pihusti lahtiolekuajaga. Ilma koormuseta pihusti lahtiolekuaja kaartidel peab olema märgitud väiksem pööretevahemik. Samuti on kaardis olevad väärtused mõnevõrra suuremad, kui eelnevalt käsitletud kaardis ja maksimumkoormusel sõites peab kütusesegu olema rikastatud [44, p. 267].



Joonis 26. Visualiseeritud 2D formaadis otsingutabel positsioonil 70DB

Definitsioonifail enamike tähtsamate otsingutabelitele on võimalik hankida TunerPro kodulehelt [46]. Antud faili kasutati ka veojõustendis katsetuste läbiviimisel.

4.4 Seadistamine ja tulemused

Katsetused viidi läbi Tallinna Tehnikakõrgkooli veojõustendis SuperFlow AutoDyn 883 AWD, mis suudab väljastada täpseid ja korduvaid mõõtetulemusi. Võimalik on läbi viia inertsteste maksimumkoormusel väljundkarakteristika saamiseks , koormusega teste mingi kindla pöördevahemiku või koormuse testimiseks, hõõrdekadude test, millega on võimalik mõõta ka siduri või rehvide läbilibisemist ning sammteste, millest arvestatakse välja pöörlevate masside kiirendamiseks vajalik võimsus [47].

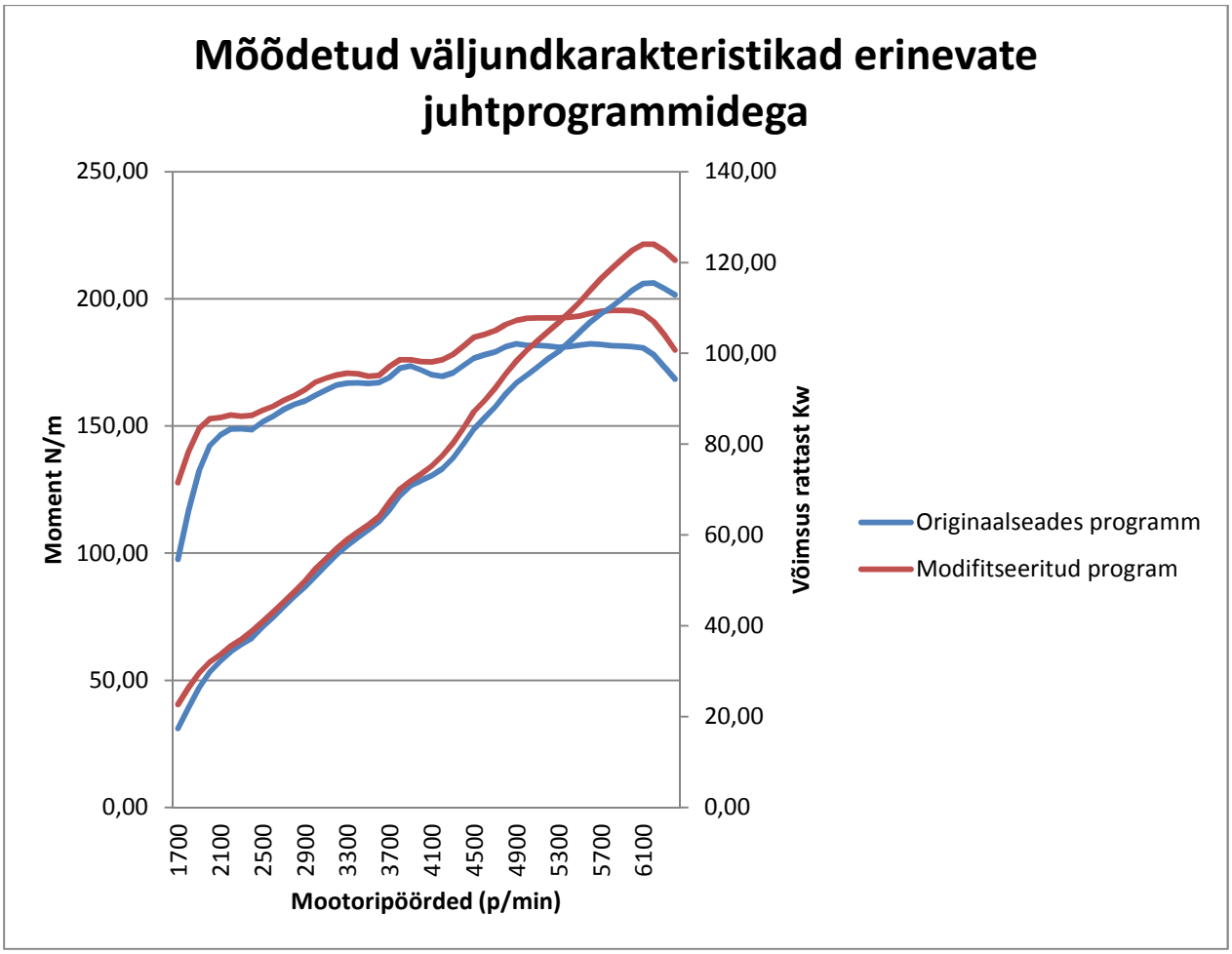
Antud töös kasutati inertsteste, kuna mõõdeti väljundkarakteristikat maksimaalse koormuse juures.

Kuna seadistamisel tuleb iga muudatuse tegemiseks mälukiip uuesti programmeerida, siis protsessi kiirendamiseks kasutati adapterit, mis tõi mälukiibi juhtarvuti korpusest välja. Kasutati survevaba pesa(inglise k. ZIF - *Zero Insertion Force*), lintkaablit ning trükkplaati, mis ühendas lintkaabli pesaga

Kütusesegu jälgimiseks kasutati lairiba lambda andurit, mida juhtis Innovate LC1 juhtplokk. Viimase väljund ühendati otse veojõustendi kontrollplokki, et logitud väärtus oleks võimalikult täpne ning sünkroniseeritud teiste parameetritega.

Detonatsiooni tuvastamiseks kasutati stetoskoopi ning kõrvaklappe.

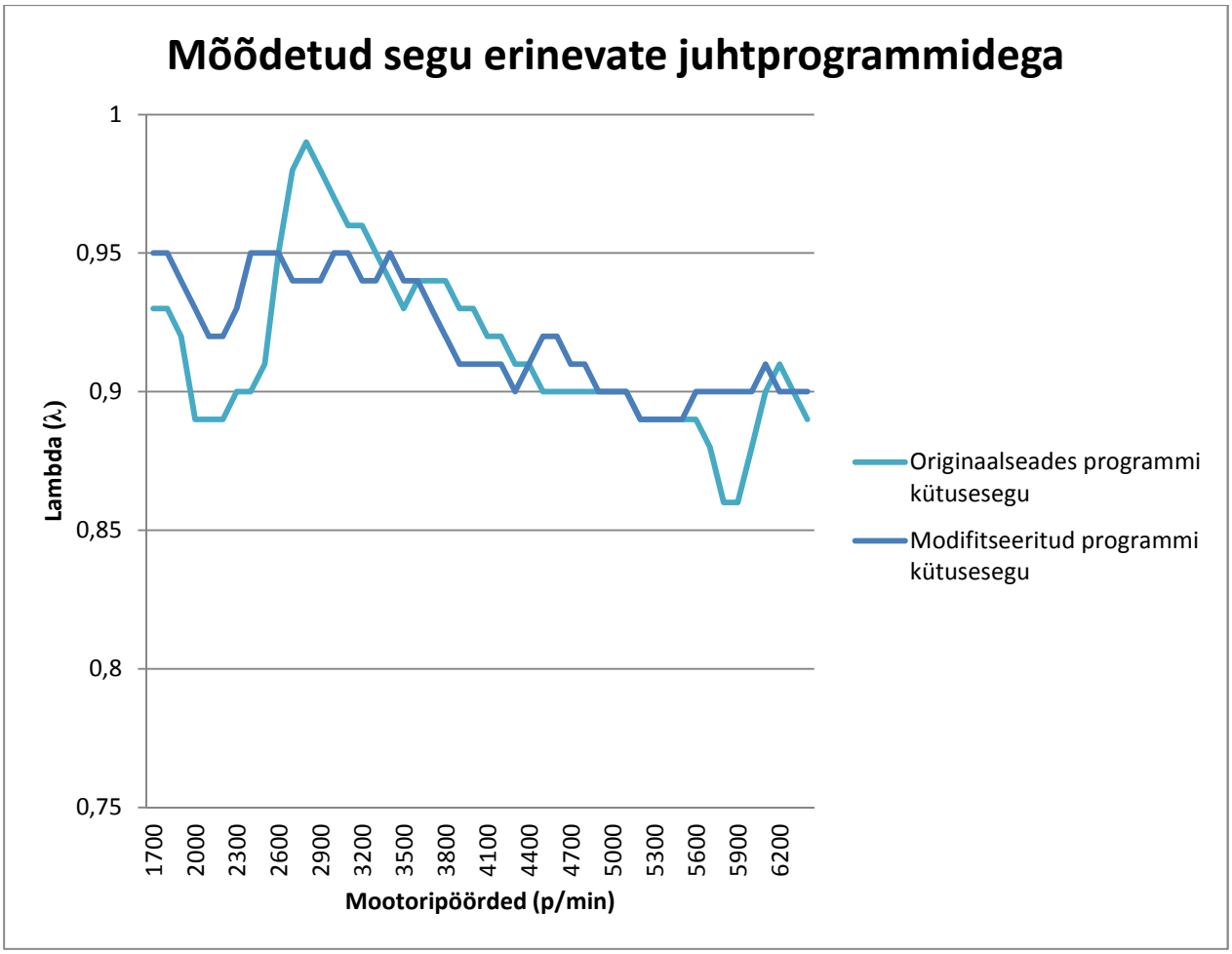
Erinevad katsed viidi läbi võimalikult samades tingimustes. Originaal juhtarvuti kasutamise tõttu tuleb igaks ümberprogrammeerimiseks mälukiip pesast eemaldada(süüde välja keeratud ning aku massiklemm maas), see programmeerida ning seejärel lasta juhtarvutil teostada adaptatsioonid. Kogu protsess võtab aega 5-7 minutit ja seetõttu ei olnud seda sõidukit seadistades probleemi, et õhutemperatuur veojõustendi ruumis läheks liialt suureks.



Joonis 27. Mõõdetud väljundkarakteristikad originaalseades programmiga ja modifitseeritud programmiga

Graafikult on näha, et nii võimsus kui ka väändemoment on kogu pööretevahemikus tõusnud (Joonis27).Selle saavutamiseks muudeti süütehetke baasväärtust maksimum koormuse korral keskmiselt 6 kraadi varasemaks võrreldes originaaliga. Prooviti ka veel varasemat süüdet – keskmiselt 8 kraadi varasem võrreldes originaaliga, kuid see enam efekti ei andnud. Detonatsiooni ei olnud kuulda kummagi ekstreemse eelsüütenurga korral. Kasutati mootoribensiini oktaaniarvuga 98.

Originaalseades, kõikus antud mootoril lambda näit märkimisväärselt, jäädes 0,99 ja 0,86 vahele.(Joonis 28)(absoluutväärtusena ei saa neid arve vaadelda, sest mõõtepingi juhtplokk ei olnud kalibreeritud töötama kasutatava lairiba juhtplokiga, kuid suhteline muut on hästi märgatav) Peale pihusti lahtiolekuaja muutmist maksimumkoormusega otsingutabelites saavutati stabiilsem segu mis jäi 0,95 ja 0.89 vahele. Katsetati ka rikkamaid segusid kuid, tulemust need ei andnud.



Joonis 28 Mõõdetud segu erinevate juhtprogrammidega

Kuna otsingutabelite resolutsioon ei ole väga suur ning vahepealsed väärtused interpoleeritakse, siis täiesti ideaalseks pole väärtusi võimalik seadistada. Interpoleerimise tõttu muutub tulemus väga suures pööretevahemikus ning seetõttu peab tegema palju katseid.

Nende tulemuste põhjal võib väita, et otsingutabeleid muutes on võimalik muuta mootori väljundkarakteristikat soovitud suunas. Kuigi antud töös modifitseeriti vaid täiskoormuse korral töötavaid otsingutabeleid, on samu meetodeid järgides võimalik optimeerida ka osakoormuse või tühikäigu otsingutabeleid.

KOKKUVÕTE

Antud lõputöös said kõik eesmärgid täidetud

Kiirussignaali saamiseks näidikuteplokki kasutati induktiiv ABS andurit ning olemasolevat ABS hammasvööd poolteljel. Kuna sellisel hammasvööl oli hammaste arv erinev ja signaalikuju teine, kui originaalis kasutataval keelreleel, siis projekteeriti signaalitöötluseks teisendusplokk. Induktiivanduri sinusoidaalne signaal teisendati LM393 komparaatori abil digitaalseks nelinurksignaaliks. See omakorda loeti mikrokontrolleri abil, mõõdeti aega eelmisest ratta täispöördest ja arvutati selle info põhjal auto kiirus. Kiirus oli aluseks väljundsignaali sageduse arvutamiseks. Väljundsignaaliks oli samuti muutuva sagedusega nelinurklaine, mille abil juhiti transistorit, mis asendas keelrelee kontakte. Kuna väljundsignaali sagedus arvutatakse mikrokontrolleris, siis on võimalik muuta programmi vastavalt kasutatavatele ratastele, seega spidomeeter näitab alati õiget kiirust. Väljundsignaali täpsus jääb 1% piiresse, mis on kiiruse puhul piisav.

Põhjalikult uuriti ning kirjeldati paralleelmälu tööpõhimõtet ning informatsiooni paiknemist mälukiibis. Saadud informatsiooni põhjal projekteeriti ümberlülitusskeem, mis võimaldab lõppkokkuvõttes muuta mootori väljundkarakteristikat. Selle saavutamiseks asendati originaalis kasutatav mälukiip kaks korda suurema mahuga mäluga ja paigutati kaks juhtprogrammi üksteise järgi. Ümberlülituseks ühelt programmilt teisele peab toimuma täpse ajastusega, mida projekteeritud lahendus arvestab. Samuti välditakse lüliti lülitamisel tekkivat müra, mis digitaalelektroonikaga kokku ei sobi. Tulevikus on sama lahendust laiendades võimalik kasutada ka rohkem informatsiooni mahutavaid mälukiipe, ümberlülitused muuta digitaalseks ja selle abil implementeerida lisafunktsioone.

Mootori väljundkarakteristika muutmiseks kirjeldati programmimälus paiknevate otsingutabelite formaati, vormingut ning viisi kuidas neid lokaliseerida ja visualiseerida. Kuna otsingutabelite funktsiooni saab ilma tehasedokumentatsioonita määrata vaid oletamise teel, siis teostati veojõustendis katsetused mille käigu optimeeriti täiskoormuse juures kasutatavaid otsingutabeleid ja saavutati keskmiselt 10% lisavõimsust. See näitas, et otsingutabelite optimeerimisega on võimalik väljundkarakteristikat muuta.

SUMMARY

Final thesis subject is „Custom Car Speed Signal Processing and Engine Control Unit Modification“. Custom car described in this thesis used components from different production cars such as BMW 525I from 1992 and Ford Sierra from 1990. This meant that not all components were incompatible with each other. For example, BMW used reed switch in differential housing for speed signal generation but Sierra deferential, witch is used on this custom car, does not have this kind of feature. The car is designed to be driven on regular streets and on track. These are totally different environments so that different sized wheels are used. This requires corrections on speed signal output circuitry so that speedometer displays correct value regardless of wheels used. Because of drastic differences of street driving and track driving it is necessary to change engine output characteristics. On street fuel economy is really important but on track only maximum performance matters. The purpose of this thesis was to find and design a solution for speed signal to replace the reed switch and change engine output characteristics when desired.

Different solutions were discussed but as the car had a anti lock braking system rotors on rear half shafts inductive sensor was used to read rotational speed. To digitize sinusoidal output of inductive sensor, LM393 comparator was used. Comparator output was square wave signal and one rising edge corresponded to one tooth on the rotor. A microcontroller was then used to calculate correct output signal based on wheels currently used. Final solution achieved accuracy error of 1%

To change the output characteristics of the engine parallel EPROM used on engine control unit was replaced with EEPROM twice the capacity. This allowed to switch between programs which were stored so that when changing logical value on one of the address pins on EEPROM the, program would change. For switching this pin with engine running, required to filter out switch bounce noise and time switching moment so that switching is performed when the memory is not accessed. Otherwise the output may become corrupt. For timing AND gate and a clocked flip-flop was used.

Changing output characteristics also required changes to the lookup tables, were base values for outputs(injector open time or moment of ignition) were stored. Thesis describes how to locate understand and visualize these tables.

Lastly engine characteristics was measured and optimized on chassis dynamometer to prove that lookup tables were located correctly. About 10% increase in output power and torque was achieved in full load condition.

All the goals set in this thesis were achieved and solutions for future improvements were described.

VIIDATUD ALLIKAD

[1] „Moore's Law," [Võrgumaterjal]. Available: http://www.mooreslaw.org/. [Kasutatud 7 märts 2014].

[2] „The device that customises car performance," [Võrgumaterjal]. Available: http://www.alfaromeo.com/com/#/technology/dna. [Kasutatud 7 Märts 2014].

[3] T. Denton, „Automobile Electrical and Electronic Systems," 2004, p. 463.

[4] Autori erakougu, 2014.

[5] „The Constituents of Semiconductor Components," [Võrgumaterjal]. Available: http://www.element14.com/community/servlet/JiveServlet/downloadBody/22518-102-1-66690/84051%20The%20Constituents%20of%20Semiconductor%20Components.pdf. [Kasutatud 10 Aprill 2014].

[6] V. Sinivee, „Elektroonika-aabits," Arvutikasutaja, kd. Veebruar, p. 13, 2005.

[7] R. Nave, „Comparator," [Võrgumaterjal]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/electronic/opampvar8.html. [Kasutatud 13 aprill 2014].

[8] R. B. GmbH, Automotive Microelectronics, 2001, p. 91.

[9] „Arduino Leonardo," [Võrgumaterjal]. Available: http://arduino.cc/en/Main/arduinoBoardLeonardo. [Kasutatud 15 aprill 2014].

[10] „8-bit Microcontroller with 16/32K Bytes of ISP Flash and USB Controller," [Võrgumaterjal]. Available: http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Summary.pdf. [Kasutatud 15 04 2014].

[11] R. B. GmbH, Automotive Sensors, 2007, p. 165.

[12] R. B. GmbH, Automotive Handbook, 2011, p. 1265.

[13] „Low Power Low Offset Voltage Dual Comparators,“ [Võrgumaterjal]. Available: http://www.wvshare.com/datasheet/NS_PDF/LM393.PDF. [Kasutatud 10 märts 2014].

[14] [Võrgumaterjal]. Available: http://www.ti.com/lit/ds/symlink/lm393.pdf. [Kasutatud 10 märts 2014].

[15] R.Nave, „Voltage Divider,“ [Võrgumaterjal]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/electric/voldiv.html. [Kasutatud 17 Märts 2014].

[16] „Standard Resistor Values (±5%),“ [Võrgumaterjal]. Available: http://ecee.colorado.edu/~mcclurel/resistorsandcaps.pdf. [Kasutatud 16 märts 2014].

[17] „SCHOTTKY BARRIER RECTIFIERS 1.0 AMPERE 20, 30 and 40 VOLTS,“ [Võrgumaterjal]. Available: http://www.onsemi.com/pub_link/Collateral/1N5817-D.PDF. [Kasutatud 16 märts 2014].

[18] „LTSPICE IV,“ [Võrgumaterjal]. Available: http://www.linear.com/designtools/software/. [Kasutatud 19 märts 2014].

[19] „Download the Arduino Software,“ [Võrgumaterjal]. Available: http://arduino.cc/en/Main/Software. [Kasutatud 25 aprill 2014].

[20] „Arduino/Processing Language Comparison,“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Comparison. [Kasutatud 15 aprill 2014].

[21] „analogWrite(),“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/AnalogWrite. [Kasutatud 6 aprill 2014].

[22] „float,“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Float. [Kasutatud 6 aprill 2014].

[23] „unsigned long,“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/UnsignedLong.

[Kasutatud 6 aprill 2014].

[24] „volatile keyword,“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Volatile. [Kasutatud 13 aprill 2014].

[25] „setup(),“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Setup. [Kasutatud 6 aprill 2014].

[26] „AttachInterrupt(),“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/AttachInterrupt. [Kasutatud 6 aprill 2014].

[27] „int,“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Int. [Kasutatud 6 aprill 2014].

[28] „millis(),“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Millis. [Kasutatud 6 aprill 2014].

[29] „map(),“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Map. [Kasutatud 7 aprill 2014].

[30] „toneAC Arduino Library,“ [Võrgumaterjal]. Available: https://code.google.com/p/arduino-tone-ac/. [Kasutatud 9 aprill 2014].

[31] „tone(),“ [Võrgumaterjal]. Available: http://arduino.cc/en/Reference/Tone. [Kasutatud 9 aprill 2014].

[32] BMW 525i, 525it, 535i M5 (E34) Electrical Troubleshooting Manual, 1992.

[33] „NPN switching transistors,“ [Võrgumaterjal]. Available: http://www.stanford.edu/class/ee133/datasheets/2n2222.pdf. [Kasutatud 15 aprill 2014].

[34] „1N4001 - 1N4007,“ [Võrgumaterjal]. Available: http://www.diodes.com/datasheets/ds28002.pdf. [Kasutatud 14 märts 2014].

[35] Microchip, „256K (32K x 8) CMOS EPROM,“ [Võrgumaterjal]. Available: http://www.jaapsch.net/psion/pdffiles/Eprom032k_datasheet_27C256.pdf. [Kasutatud 16 aprill 2014].

[36] I. C. E. CORPORATION, „ROM, EPROM, AND EEPROM TECHNOLOGY,“ [Võrgumaterjal]. Available: http://smithsonianchips.si.edu/ice/cd/MEMORY97/SEC09.PDF. [Kasutatud 21 aprill 2014].

[37] S. S. Technology, „256 Kbit / 512 Kbit / 1 Mbit / 2 Mbit (x8) Many-Time Programmable Flash,“ [Võrgumaterjal]. Available: http://www.futurlec.com/Memory/SST27SF512.shtml. [Kasutatud 16 aprill 2014].

[38] „Single Positive-Edge-Triggered D-Type Flip-Flop,“ [Võrgumaterjal]. Available: http://www.ti.com/lit/ds/symlink/sn74lvc1g80.pdf. [Kasutatud 16 märts 2014].

[39] NXP, „BC847 series,“ [Võrgumaterjal]. Available: http://www.nxp.com/documents/data_sheet/BC847_SER.pdf. [Kasutatud 28 märts 2014].

[40] NXP, „BC856; BC857; BC858,“ [Võrgumaterjal]. Available: http://www.nxp.com/documents/data_sheet/BC856_BC857_BC858.pdf. [Kasutatud 28 märts 2014].

[41] „MM74HC175 Quad D-Type Flip-Flop With Clear,“ [Võrgumaterjal]. Available: https://www.fairchildsemi.com/ds/MM/MM74HC175.pdf. [Kasutatud 28 aprill 2014].

[42] F. Wilk, „Maps - 3D Frmated Maps,“ [Võrgumaterjal]. Available: http://motronic.ws/map301.htm. [Kasutatud 10 veebruar 2014].

[43] M. Mansur, „TunerPro and TunerPRO RT,“ [Võrgumaterjal]. Available: http://www.tunerpro.net/downloadApp.htm. [Kasutatud 28 veebruar 2014].

[44] A. G. Bell, Four-Stroke Perfomance Tuning, 2006, p. 479.

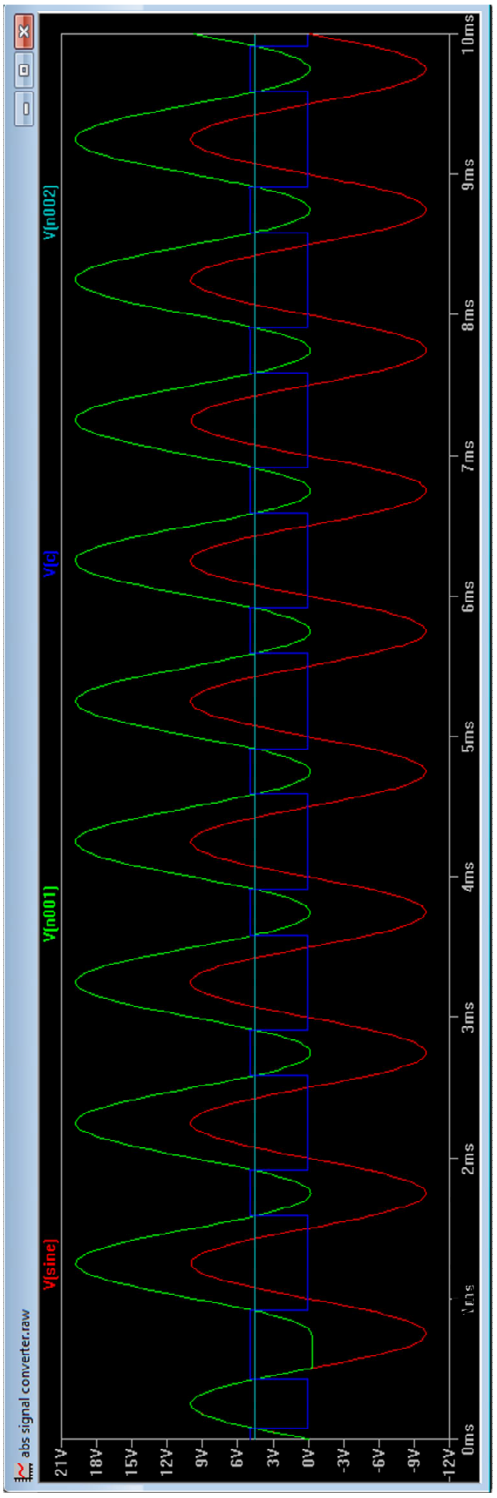
[45] F. Wilk, „RPM-Engine RPMs,“ [Võrgumaterjal]. Available: http://www.motronic.ws/rpm.htm. [Kasutatud 10 veebruar 2014].

[46] „BMW DME 405,“ [Võrgumaterjal]. Available: http://www.tunerpro.net/download/bindefs/BMW/BMW_dme405_soft951.xdf. [Kasutatud 15 veebruar 2014].

[47] T. Tehnikakõrgkool, „Spetsifikatsioon,“ [Võrgumaterjal]. Available: http://wwwold.tktk.ee/?id=1850. [Kasutatud 30 aprill 2014].

[48] NKK, „Miniature Toggles,“ [Võrgumaterjal]. Available: http://www.farnell.com/datasheets/70412.pdf. [Kasutatud 29 märts 2014].

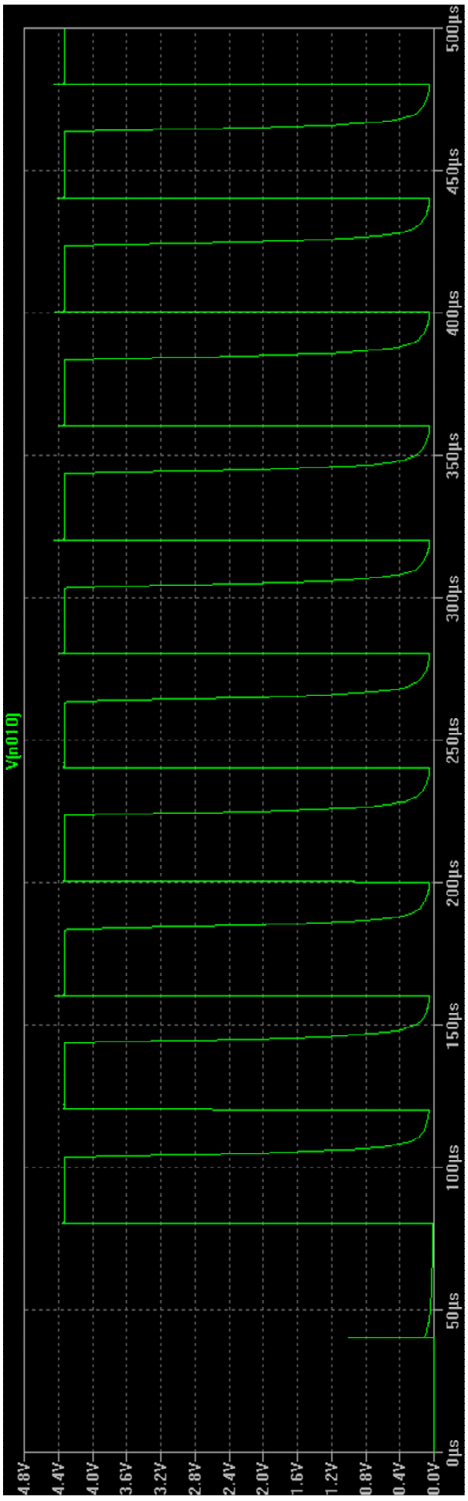
LISA 1. ABS anduri signaali teisendamine nelinurklaineks



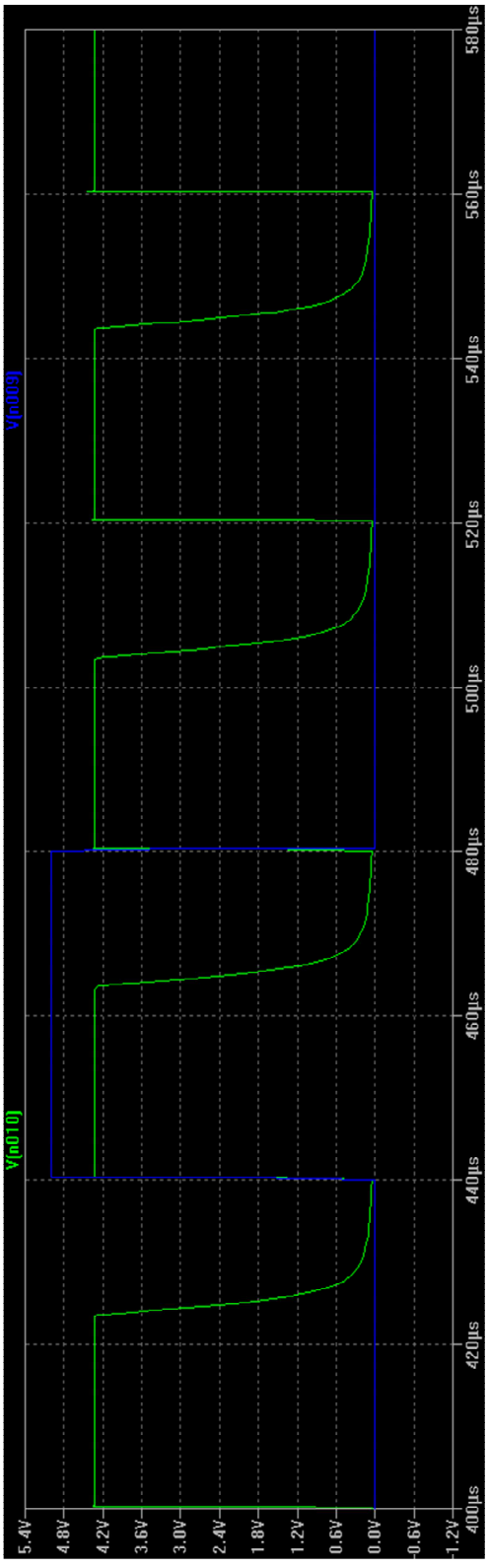
Märkused. 1. Sinine -komparaatori väljundsignaal

2. Punane -ABS andurist tulev signaal
3. Roheline -komparaatori positiivses sisendis asuv signaal
4. Helesinine -komparaatori negatiivses sisendis asuv signaal
5. hüstereesiks vt joonis 8

LISA 2. Väljundisignaal NING lülitusest(trigeri CLK sisend)



LISA 3. Väljundi muutumine CLK signaali tõustes



Märkus. 1. Roheline - CLK sisend

2. Sinine -Sinine trigeri väljund

LISA 4. Väljundi muutumise ajastus lüliti lülitamisel

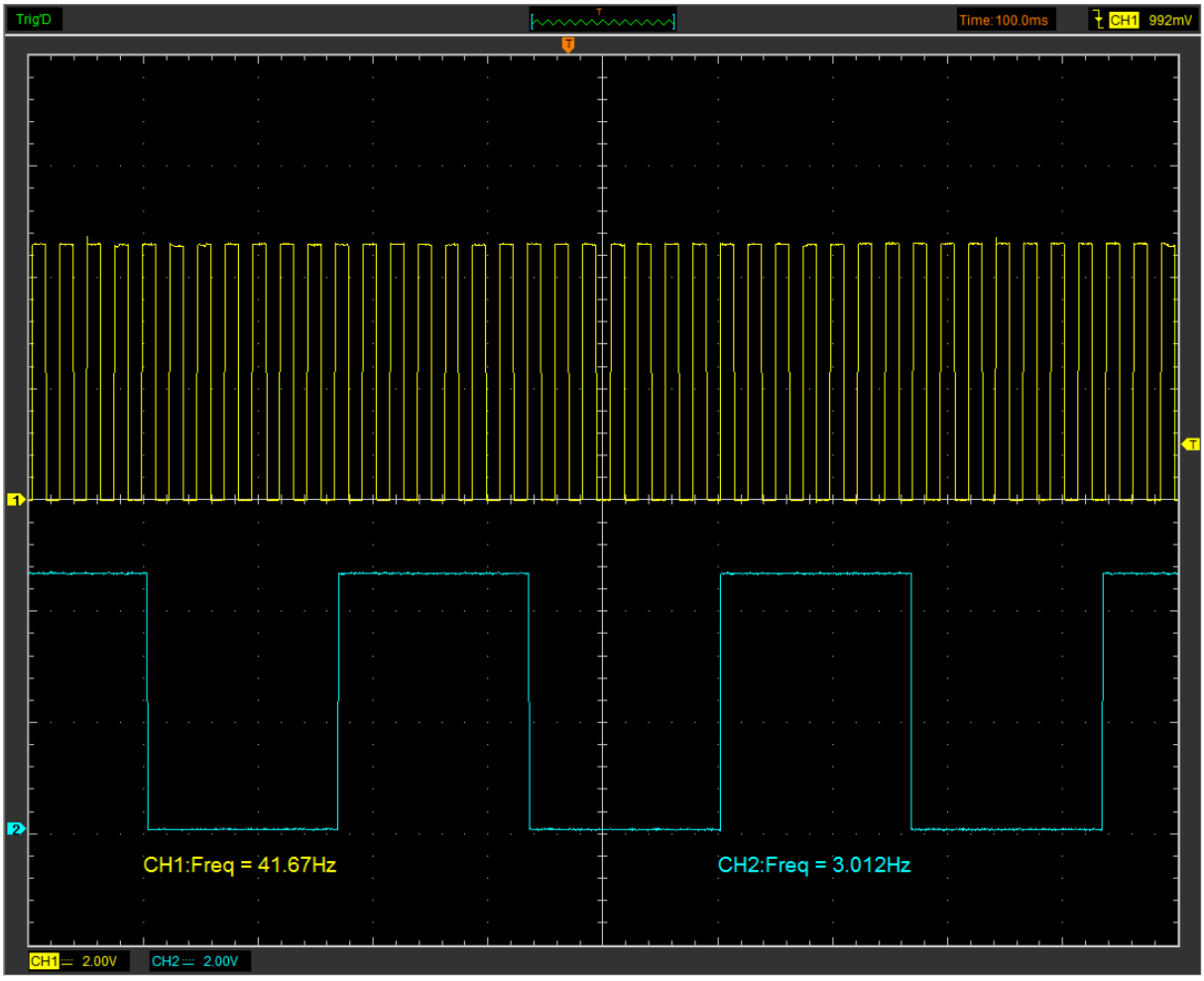
Märkused.

1. Sinine -väljudsignaal (mälukiibi aadressiviik)
2. Punane -lüliti signaal
3. Roheline -taktisagedussignaal CLK
4. Reaalselt pole sellise kiirusega võimalik lüliti olekut muuta, kuid skeemi tööpõhimõte jääb samaks
5. Lüliti signaali varajane tõus on tingitud mõlema lüliti samaaegsel vajutamisel, mis hoovaga lülitit kasutades pole reaalselt võimali

LISA 5. Programmi kood

```
#include <toneAC.h>
// Constants
const int TREAD_WIDTH = 195; // mm
const int ASPECT_RATIO = 60;   // relation
const int RIM_SIZE = 15;     // inches

const int ABS_TOTAL_TEETH = 80;
const int SPEED_OUT_PIN = 9;
const int SPEED_IN_INTERRUPT = 0; // 0 for digital pin 4

const float TIRE_CIRC = ((TREAD_WIDTH * (ASPECT_RATIO / 100.0) * 2) + (RIM_SIZE * 25.4)) * M_PI; // mm

volatile unsigned long prev_time = 0;
volatile int counter = 0;

void setup() {
//  Serial.begin(9600); // DEBUG
  pinMode(SPEED_OUT_PIN, OUTPUT);
  prev_time = millis();
  attachInterrupt(SPEED_IN_INTERRUPT, countUp, RISING);
}

void loop() {
  if(counter >= ABS_TOTAL_TEETH) {
    unsigned long current_time = millis();
    int time_delta = current_time - prev_time;
    prev_time = current_time;
    counter = 0;

    float velocity = TIRE_CIRC / time_delta;

    int vspeed = map((int)velocity * 100, 0, 7000, 0, 252);

    //Serial.println(vspeed); // DEBUG
    toneAC(vspeed);
  }
}

void countUp() {
  counter++;
}
```

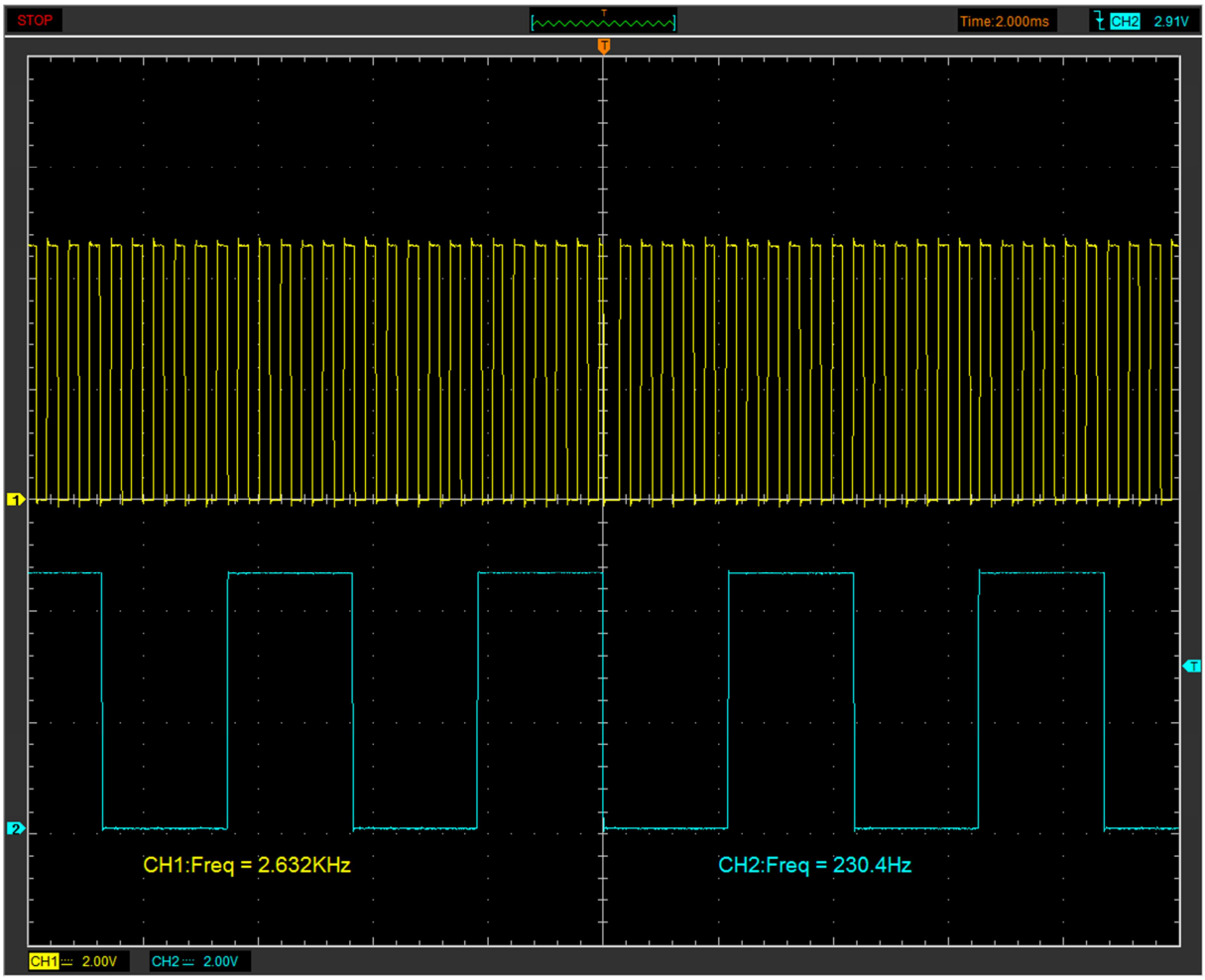
LISA 6. 195/60 R15 rehviga väljundsagedus arvutuslikul kiirusel 3,61km/h



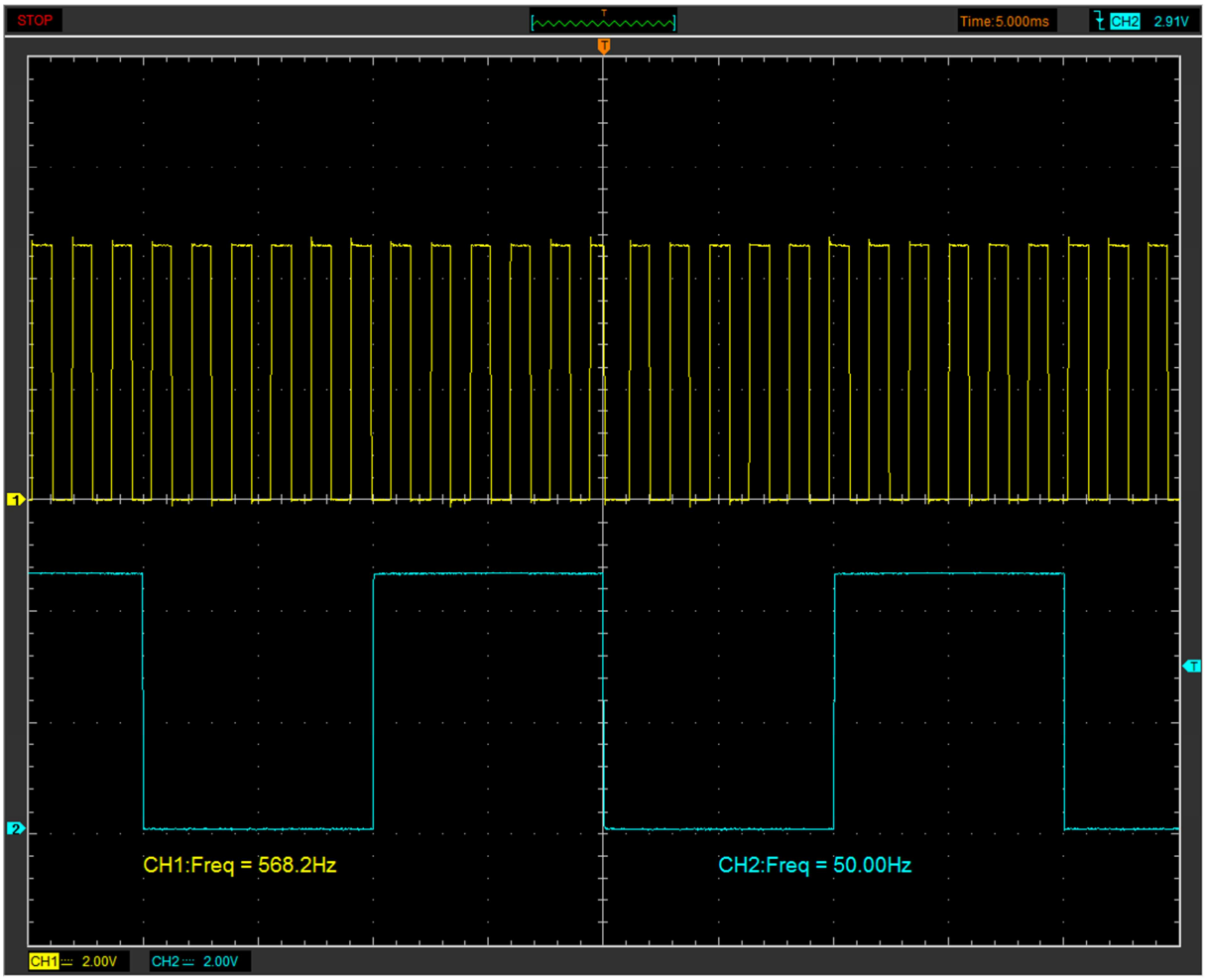
Märkus.

1. Mõõdetud kiirus 3,012 km/h, arvutuslik 3,61km/h
2. Sagedust mõõdetakse ekraanil oleva info põhjal. Ostsilloskoop ei oma raudvaralist sageduslugejat, seega tulemused võivad reaalsusest mõnevõrra erineda.
3. Suurema rehviga on sellise kiiruse juures väljundsagedus mõõdetuna täpselt sama, arvutuslik kiirus on 3,85km/h

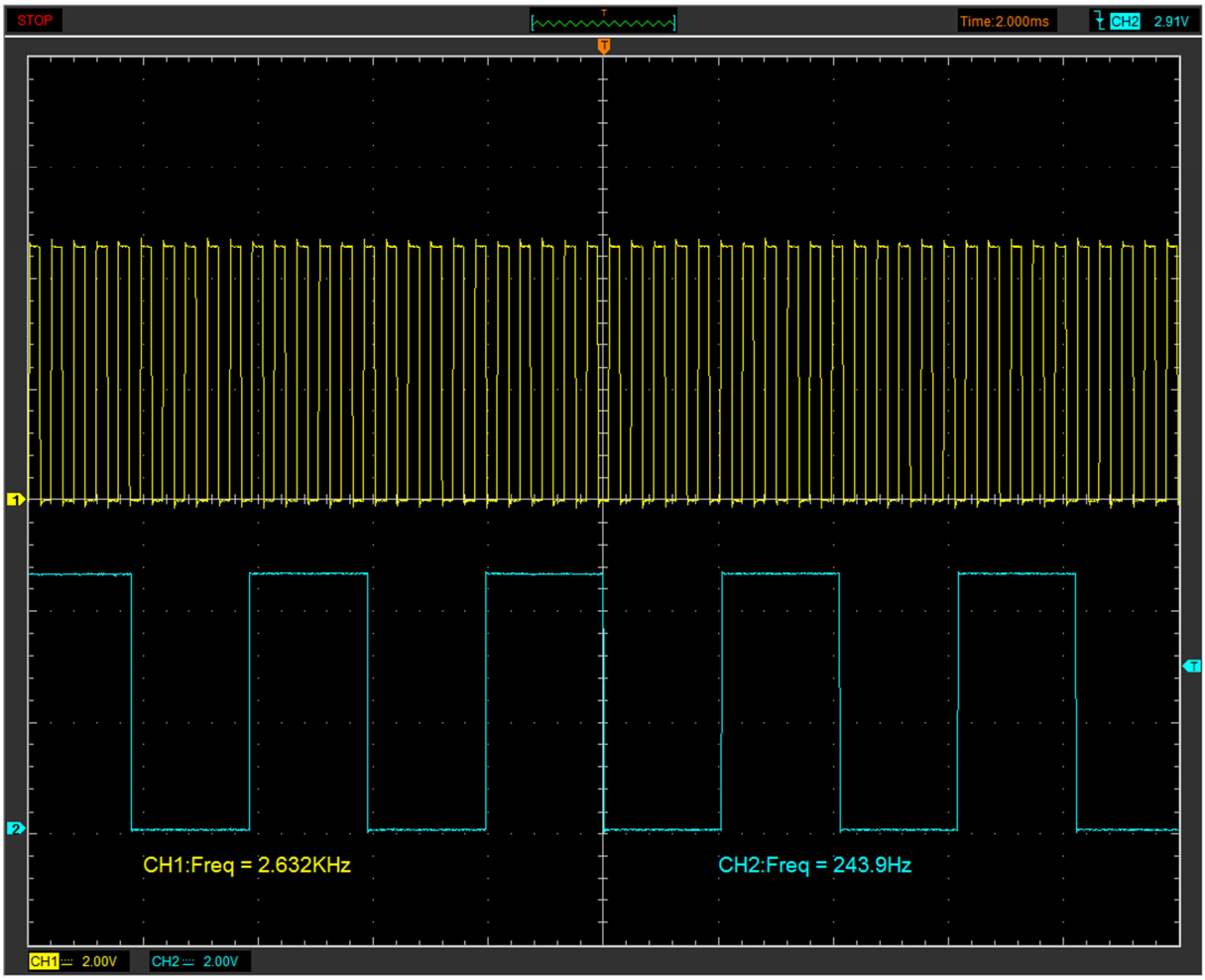
LISA 7. 195/60 R15 rehviga väljundsagedus arvutuslikul kiirusel 228,48km/h



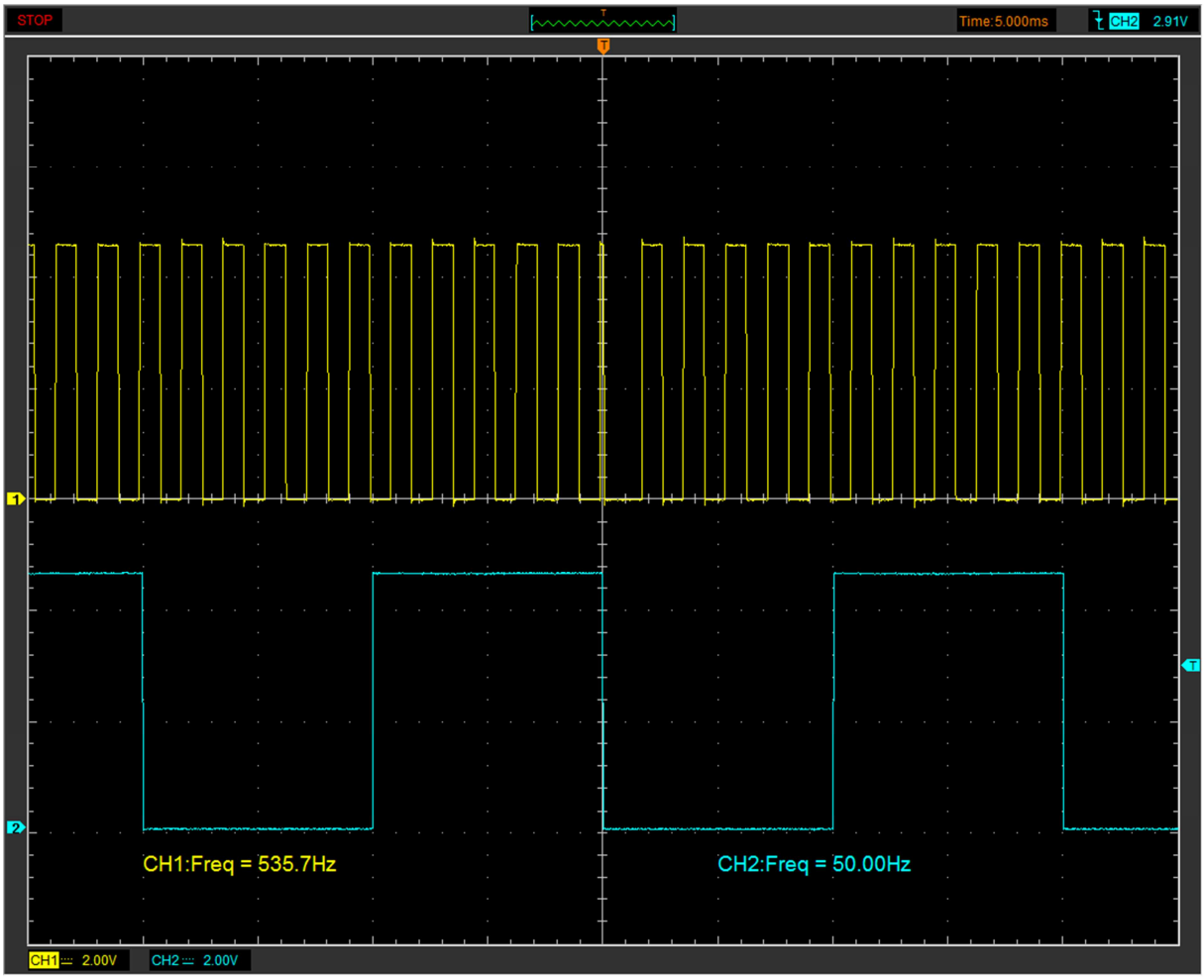
LISA 8. 195/60 R15 rehviga väljundsagedus arvutuslikul kiirusel 49,32km/h



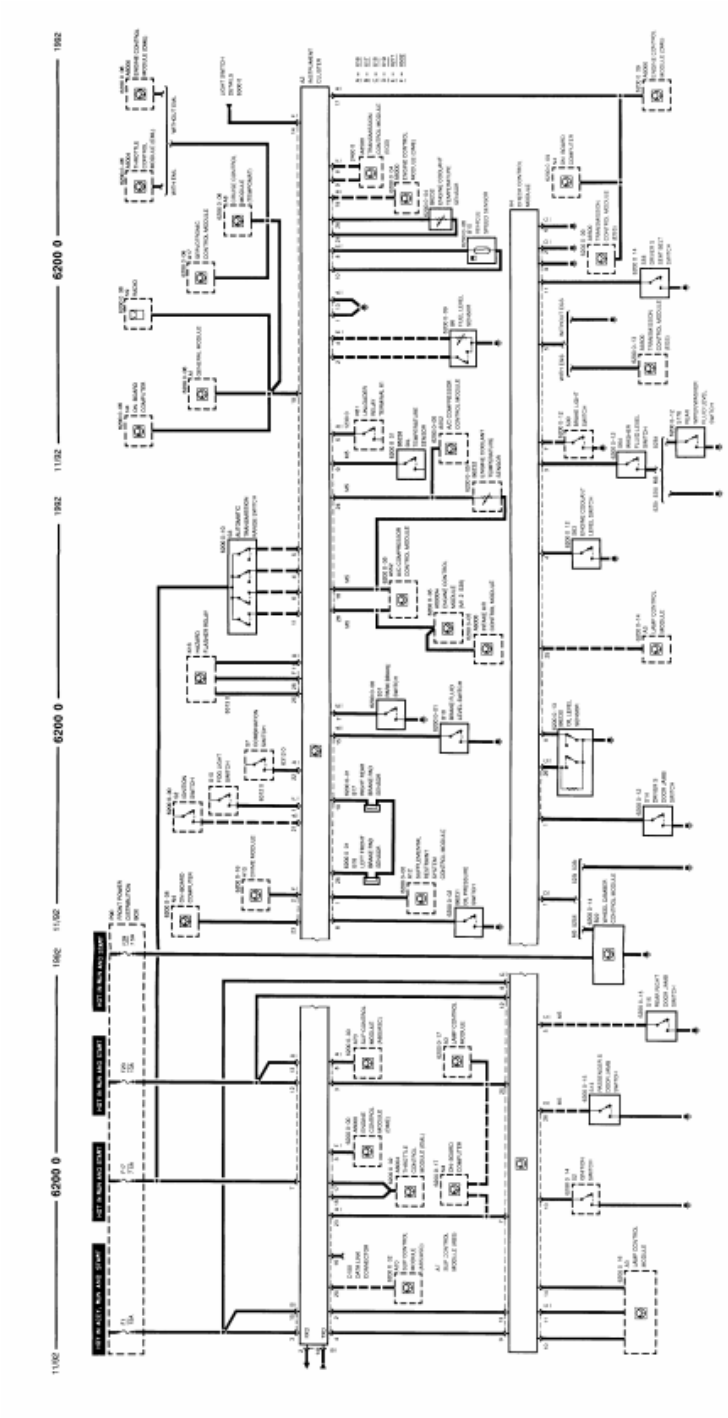
LISA 9. 245/40 R18 rehviga väljundsagedus arvutuslikul kiirusel 243,8km/h



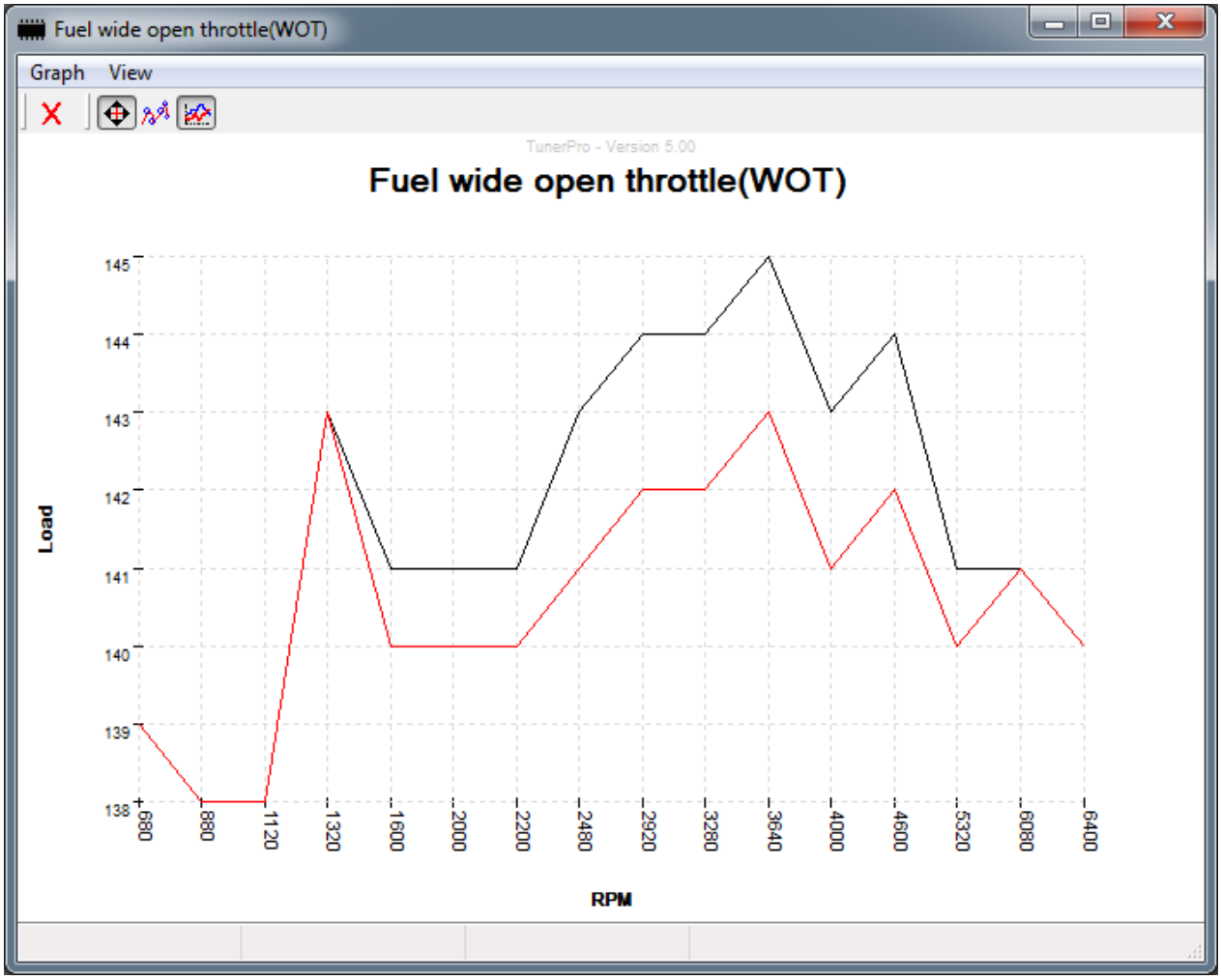
LISA 10. 245/40 R18 rehviga väljundsagedus arvutuslikul kiirusel 49,62 km/h



LISA 11. Väljavõte BMW näidikuteploki elektriskeemist



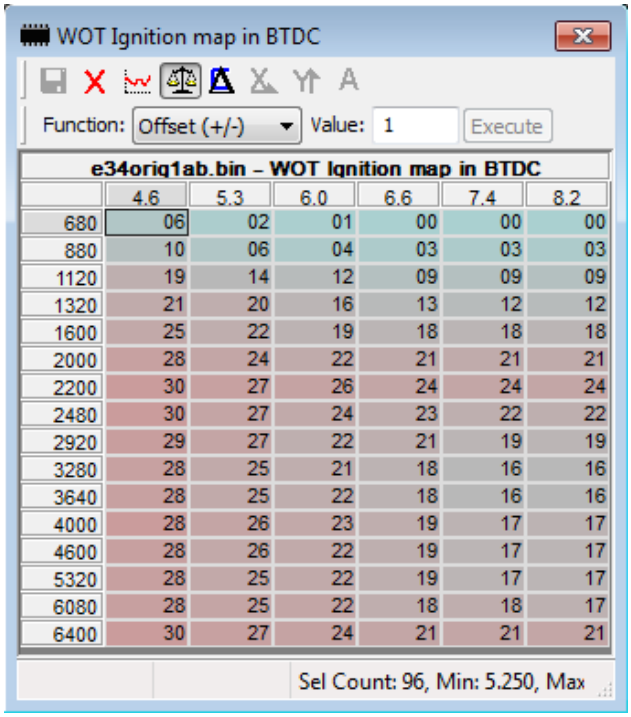
LISA 12. 100% koormusega pihusti lahtiolekuaeg (originaal vs modifitseeritud)



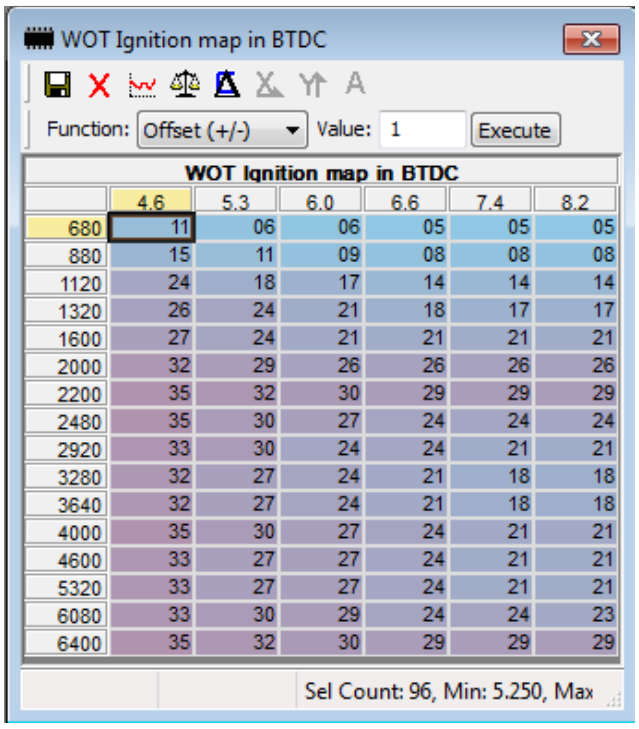
Märkused. 1. Punane -originaal

2. Must -modifitseeritud

LISA 13. 100% koormusega süütehetk enne ülemist surnud seisu (originaal vs modifitseeritud)



	4.6	5.3	6.0	6.6	7.4	8.2
680	06	02	01	00	00	00
880	10	06	04	03	03	03
1120	19	14	12	09	09	09
1320	21	20	16	13	12	12
1600	25	22	19	18	18	18
2000	28	24	22	21	21	21
2200	30	27	26	24	24	24
2480	30	27	24	23	22	22
2920	29	27	22	21	19	19
3280	28	25	21	18	16	16
3640	28	25	22	18	16	16
4000	28	26	23	19	17	17
4600	28	26	22	19	17	17
5320	28	25	22	19	17	17
6080	28	25	22	18	18	17
6400	30	27	24	21	21	21



	4.6	5.3	6.0	6.6	7.4	8.2
680	11	06	06	05	05	05
880	15	11	09	08	08	08
1120	24	18	17	14	14	14
1320	26	24	21	18	17	17
1600	27	24	21	21	21	21
2000	32	29	26	26	26	26
2200	35	32	30	29	29	29
2480	35	30	27	24	24	24
2920	33	30	24	24	21	21
3280	32	27	24	21	18	18
3640	32	27	24	21	18	18
4000	35	30	27	24	21	21
4600	33	27	27	24	21	21
5320	33	27	27	24	21	21
6080	33	30	29	24	24	23
6400	35	32	30	29	29	29

LISA 14. Töös kasutatud üksikorras valmistatud sõiduk