Tartu Ülikool
Füüsika-keemiateaduskond

Martin Vahi

MIKROPROTSESSOR "ÜKSBITT" SÜDAMIK

Õppeaine "Mikroprotsessorid" kodutöö

Tartu 2002/2003

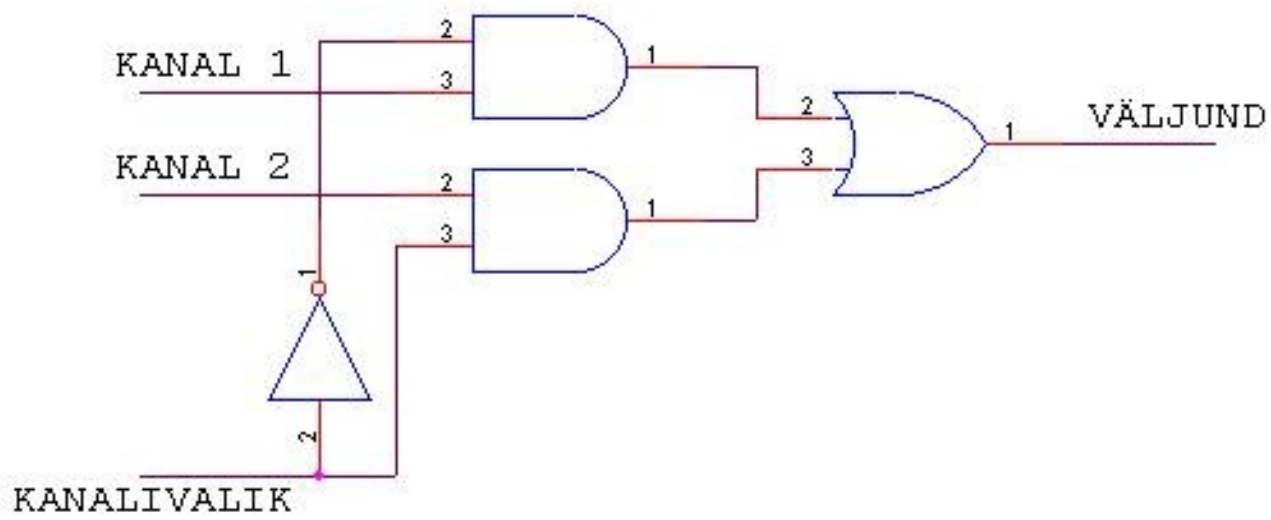
SISUKORD

SISSEJUHATUS

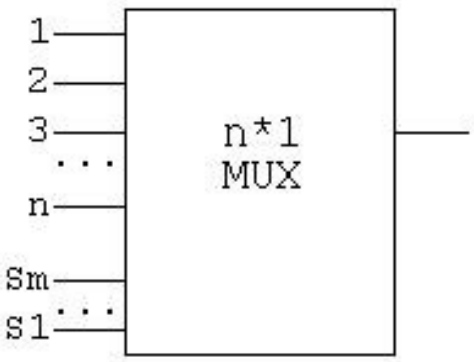
Kuna tetgu on väga mahuka ülesandega, siis ei ole selles kirjatükis reaalset skeemi toodud, vaid on detailselt ära kirjeldatud kõik tükid, millest reallse skeemi kokku saab panna. Protsessori nimetus on küll “Üksbitt”, kuid tegelikult on tegu siiski 8-bitilise protsessoriga. Seoses sellega, et see töökene on kirjutatud Microsoft Wordiga, kus ei ole korralikke vahendeid jooniste numeratsiooni koostamiseks, ei vasta kahjuks jooniste numeratsioon nende esinemisjärjekorrale tekstis.

LOOGLINE MULTIPLEKSOR

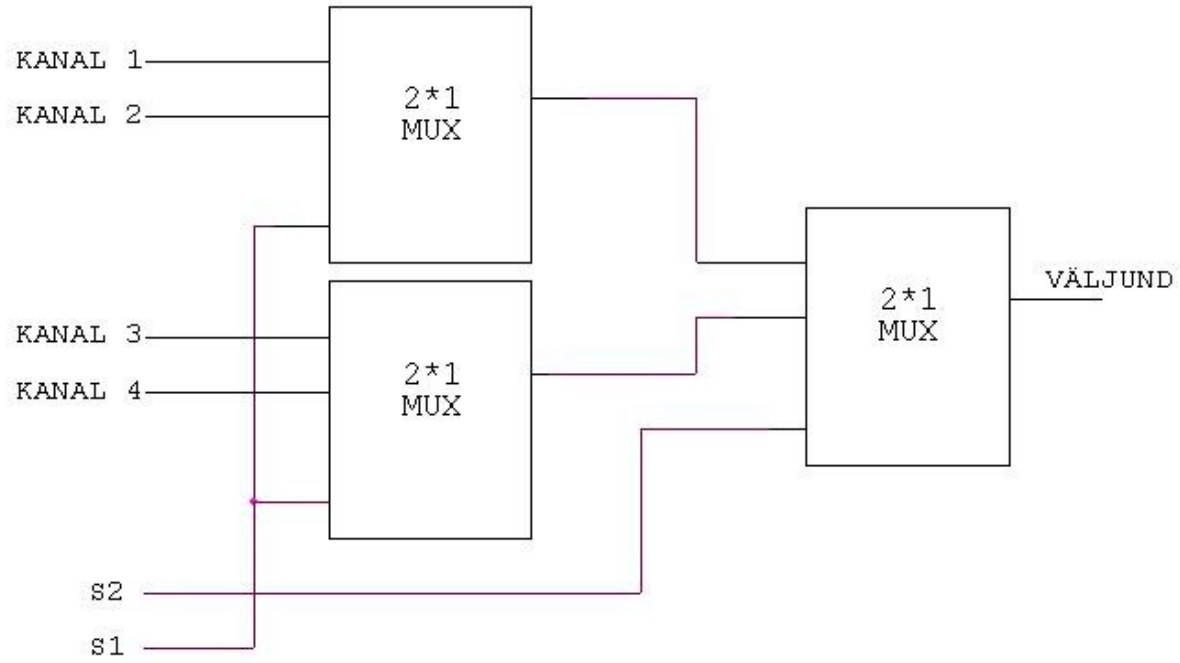
Järgnevalt vaatlen ainult loogilistele signaalidele mõeldud multipleksoreid ja seda Boole´i loogika järgi, st. et antud juhtumil 3. kõrgeimpedantsilist olekut ükski skeemielement kunagi ei oma.



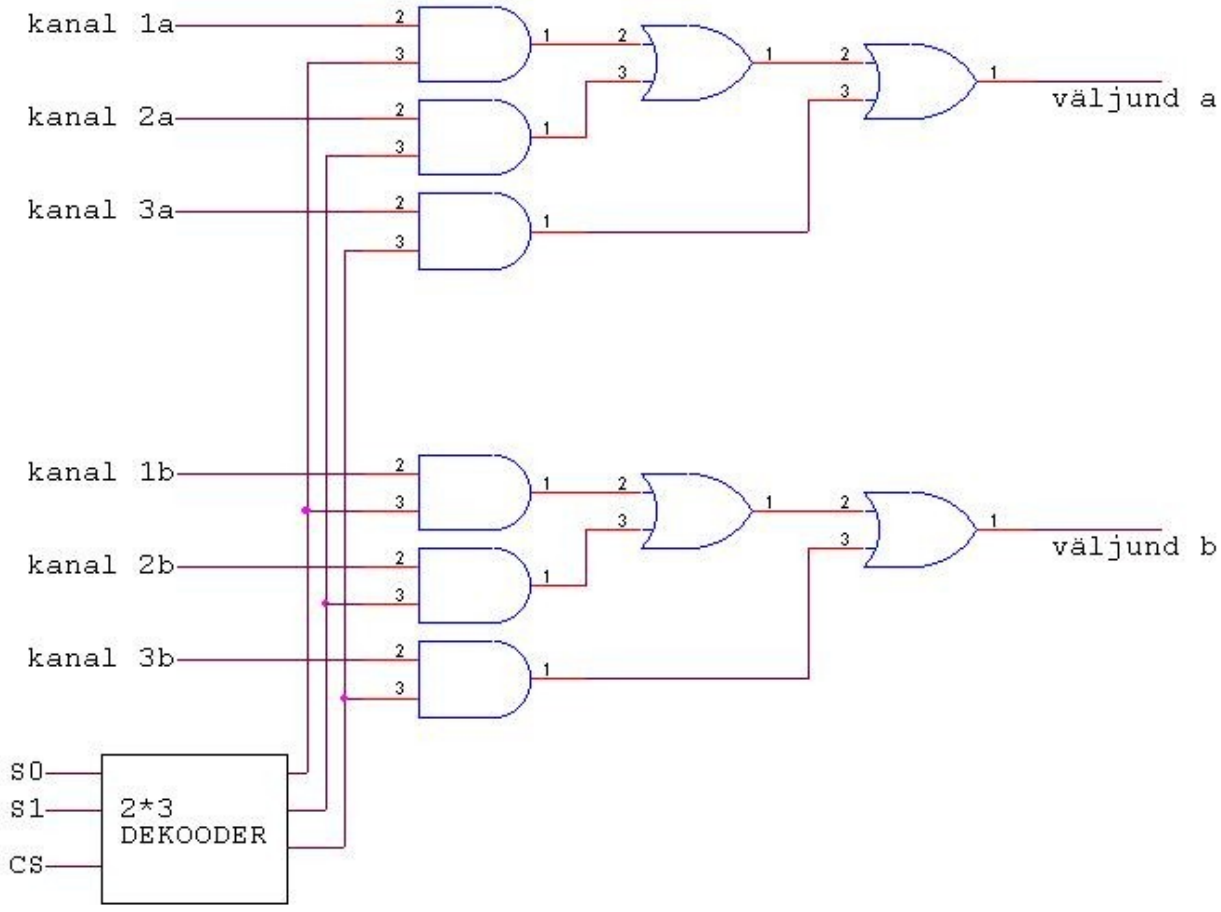
Joonis # 20



Üheväljundilise ilma CS-viiguta multipleksori üldtähistus.
Viigud Sx on kanalite valikuks.
Joonis # 21



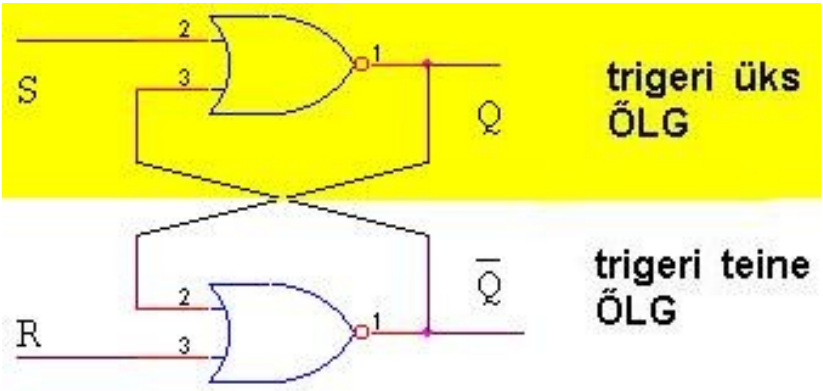
4*1 MUX
Joonis # 22



näide dekoodrit kasutavast mitmväljundilisest multipleksorist
Joonis # 27

TRIGERITE KIRJELDUSI

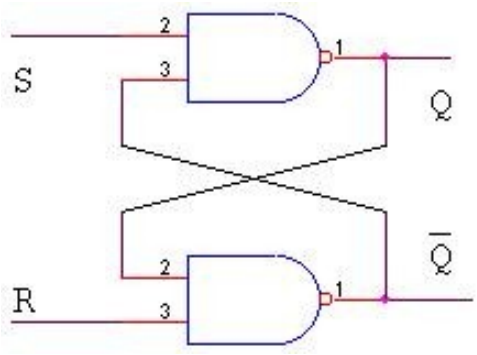
Kuna protsessori paljud osad koosnevad just trigeritest, siis proovin järgnevalt anda lühikirjelduse osadest “Üksbitt´is” rakendustleidvatest trigeritest. Kasutan oma tekstis väljendit “trigeri õlg”, mille all ma mõtlen trigeri skeemist ühte poolt, kui see skeem mõttes sümmeetriliselt pooleks teha(joonis #1). Trigeri sisendiks nimetan ma S-viigule ja R-viigule antud väärtuste paari ning trigeri väljundi all mõtlen ma Q-viigust ja mitteQ-viigust saadud väärtuste paari. Ühe õla väljundi all mõtlen ma kas Q-viigust või mitteQ-viigust saadud väärtust. Ühe õla sisendiks loen ma kas S-viiku või R-viiku antud väärtust. Mitte-SR-trigerite puhul on kasutan ma S-viigu ja R-viigu asemel muid vastavaid viikude nimetusi.



joonis #1

SR-triger

SR-trigeri üks loogiline skeem on kujutatud joonisel #1. Kuna eksisteerib ka teine SR-trigeri loogiline skeem(joonis #2), siis nimetan joonisel #1 kujutatud skeemi 1. tüüpi SR-trigeriks ja joonisel #2 kujutatud skeemi 2. tüüpi SR-trigeriks. 1. tüüpi SR-trigeri olek säilib sisendi 00 korral ning keelatud sisendi 11 korral on väljund 00. 1. Tüüpi SR-trigeri ühe õla sisendi seadmine 1´ks seab selle õla väljundi 0´ks.

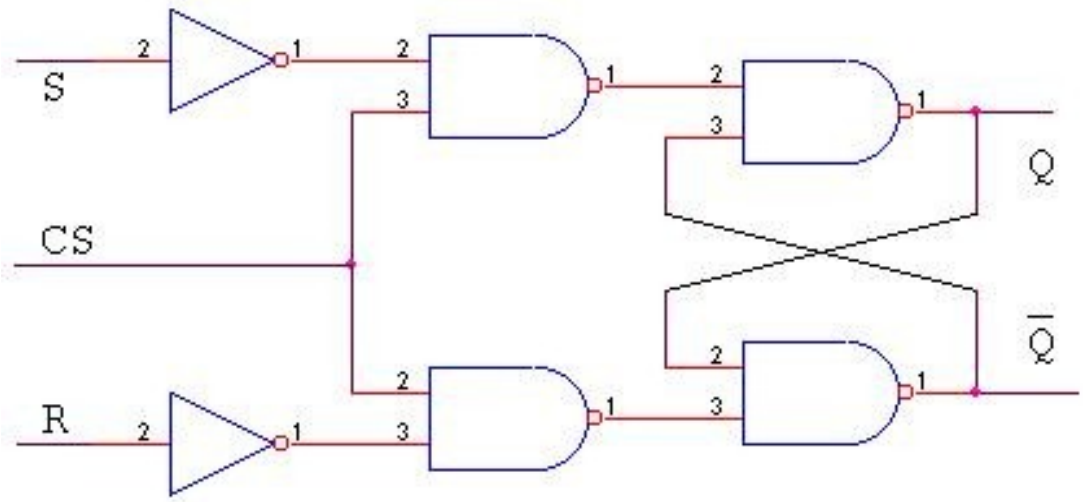


joonis # 2

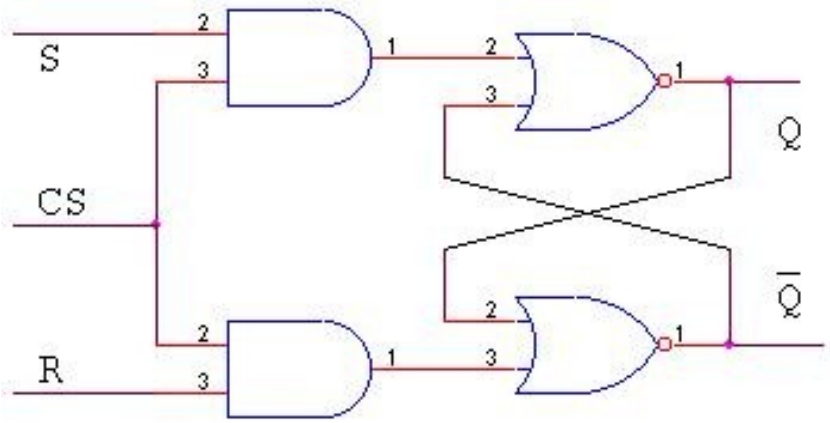
2. tüüpi SR-trigeri väärtus säilib sisendi 11 korral ning keelatud sisendväärtuse 00 korral on väljundiks 11. Ühe õla sisendi muutumisel 0´ks muutub selle õla väljund 1´ks.

Loogiliselt erinevad 1. tüüpi ja 2. tüüpi trigerid üksteisest ainult selle poolest, et ühe keelatud sisendväärtus on teise säilitamisväärtus ja vastupidi.

Joonisel # 10 on kujutatud 2. tüüpi SR-trigeriga loogiliselt identne takteeritud triger ning joonisel # 11 on kujutatud 1. tüüpi SR-trigeriga loogiliselt identne takteeritud triger.



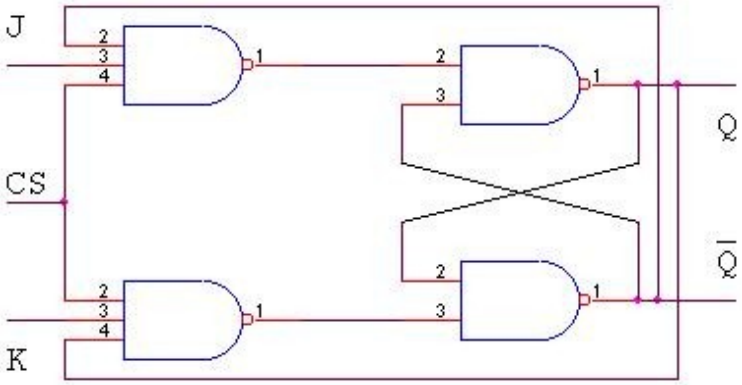
Joonis # 10



Joonis # 11

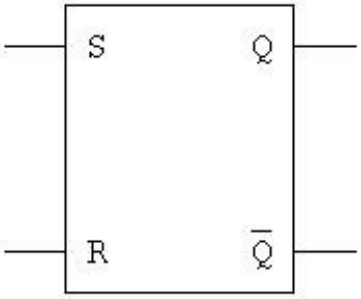
JK-triger

JK-trigerile sisendi 00 andmine on samaväärne CS-viigule 0-väärtuse andmisega, mille puhul JK-trigeri väärtus säilib. Keelatud sisendväärtuseks on 11, mille puhul on ka väljund 11. Õla sisendisse 1 andmine seab selle õla väljundi 1´ks.

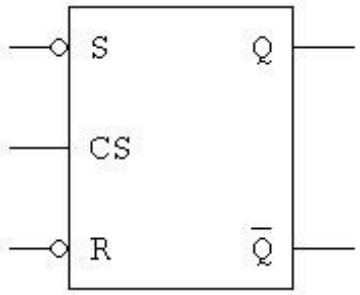


joonis # 3

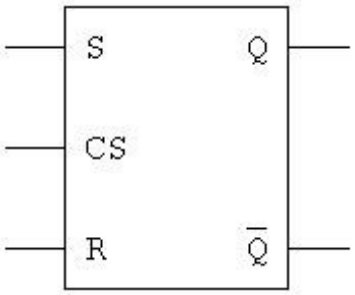
Edaspidi kasutan joonisel # 4 näidatud tähistusi.



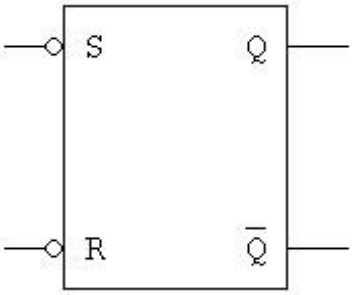
1. tüüpi SR-triger



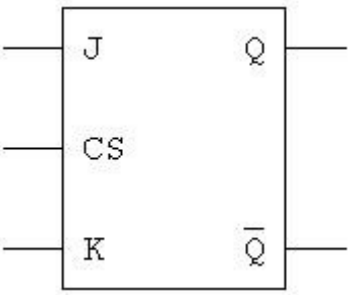
takteeritud 2. tüüpi SR-triger
(See on minu omavoliline täihistus, sest ma ei leidnud mujalt sellekohast tähistust.)



takteeritud 1. tüüpi SR-triger



2. tüüpi SR-triger

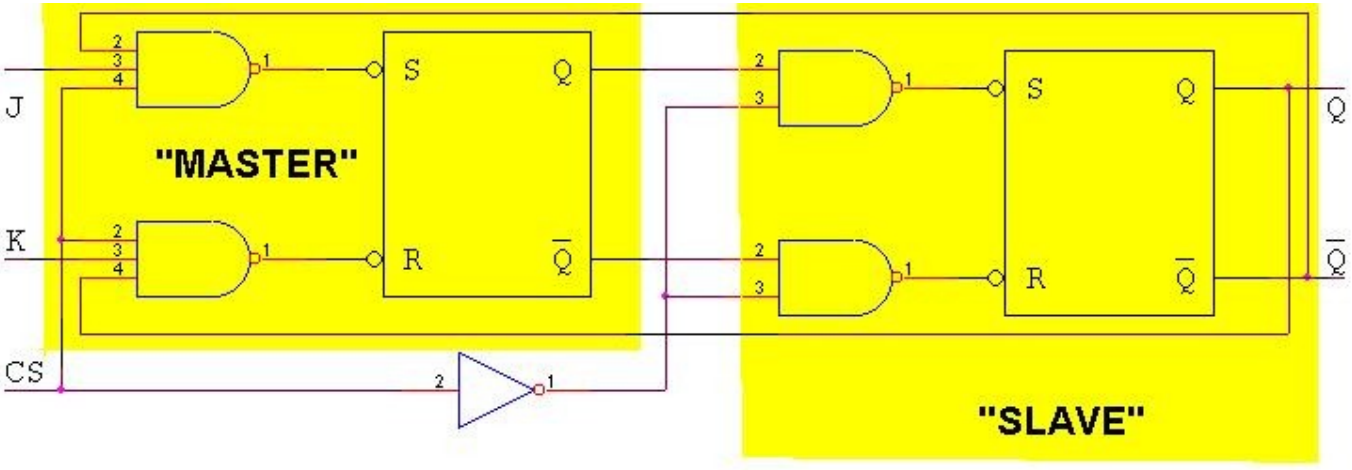


JK-triger

Joonis # 4

MS-triger

MS-triger(Master-Slave Flip-Flop) on kujutatud joonisel # 6.



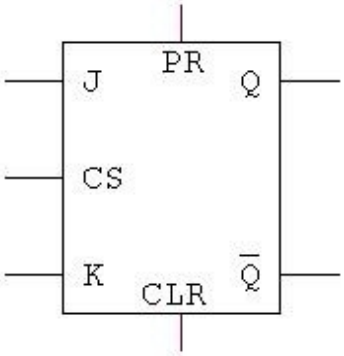
NB! Joonisel olevad SR-trigrid on “2. tüüpi SR-trigerid”.

Joonis # 6

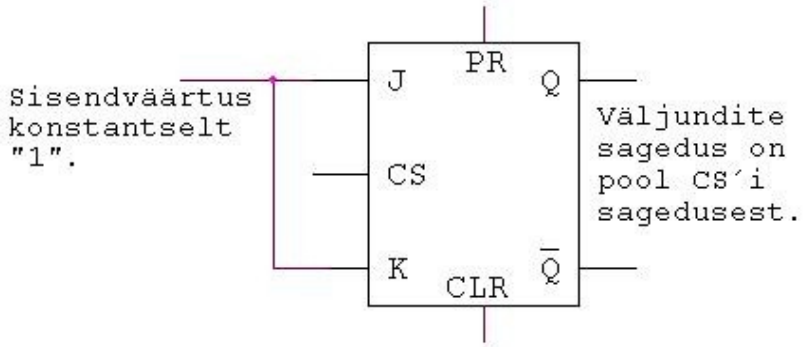
Kui joonisel # 6 kujutatud MS-trigeri sisend(J ja K viigud) on konstantselt 11, siis iga CS-viigu madala poolperioodi jooksul muutub MS-trigeri väljund täpselt vastupidiseks(inverteerub). CS-viigu madala poolperioodi jooksul kandub MS-trigeri “MASTER-osa” väärtus üle MS-trigeri “SLAVE-osasse”. Kui anda MS-trigeri sisendisse väärtus 00, siis “MASTER-osas” asuv SR-triger säilitab oma väärtuse sõltumata CS-väärtusest ning MS-trigeri väljund CS-viigu väärtuse muutumisest ei sõltu, st. on ka konstantselt kas 10 või 01. Kui joonisel # 6 kujutatud MS-tüüpi JK-trigeri sisendisse anda väärtus 10 või 01, siis sarnaselt tavalise JK-trigeriga muutub õla väljund, mille sisendisse anti 1, ka 1 ´ks, ainult et seda hiljemalt taktsignaali 1 poolperioodipikkuse viivitusega.

MS-tüüpi JK-trigerit tähistatakse tihtipeale nii, nagu on toodud joonisel # 8. Tähistusel olevad ühendid “PR” ja “CLR” viitavad sõnadele “Preset”(set) ja “Clear”(reset). Joonisel # 23 oleva MS-tüüpi JK-trigeri puhul seatakse “CLR” ja “PR” viikude kaudu trigeri “SLAVE” osa väljundit CS-signaali kõrgel poolperioodil.

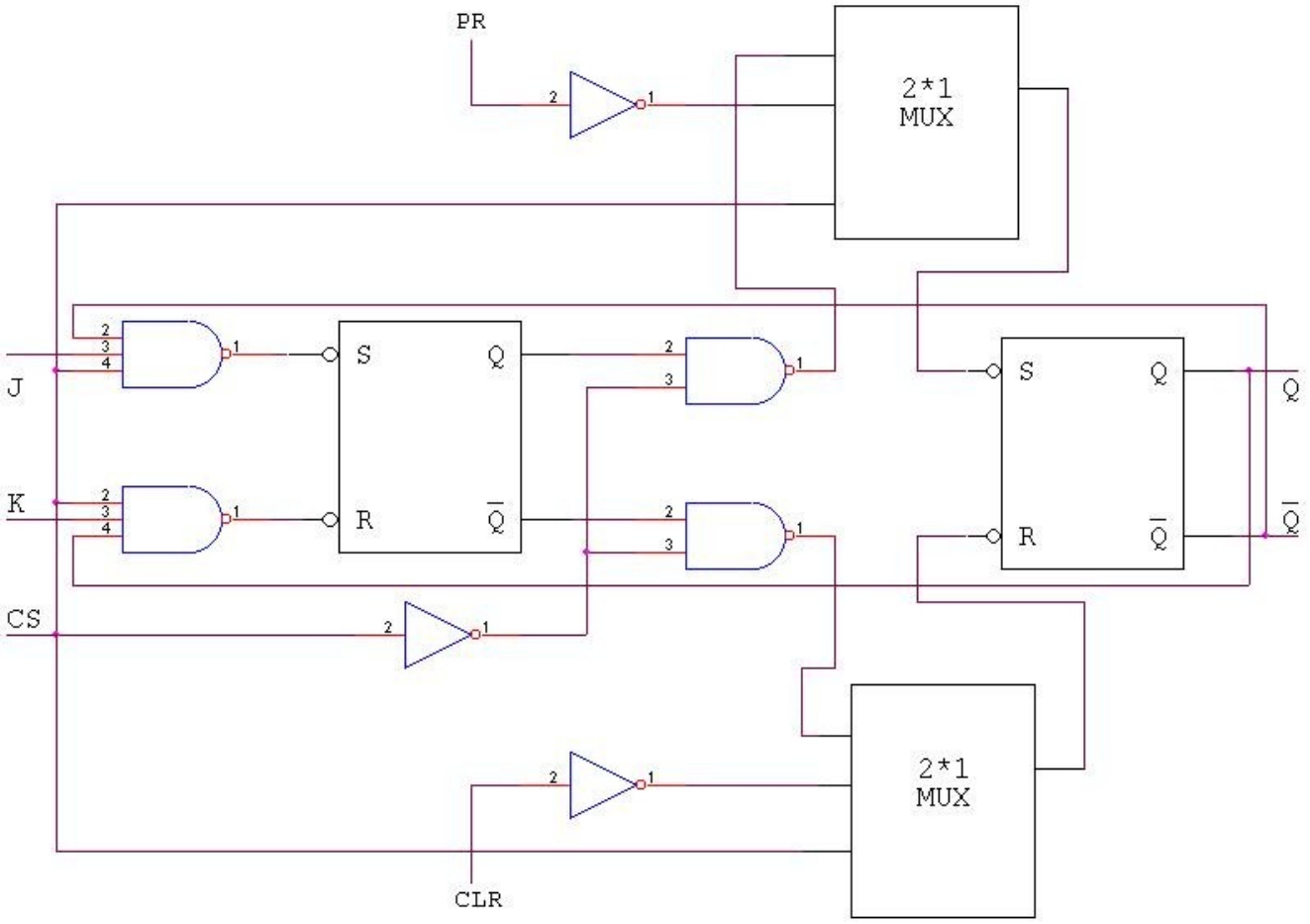
Takteerimissignaali sagedust saab jagada kahega andes MS-tüüpi JK-trigeri sisendisse konstantse signaali 11(joonis # 9). Joonisel # 12 kujutatud skeemi saab kasutada taktsignaali jagamiseks 16-ga kui ka kahendloendurina. Kahendloenduri puhul saab CLR-viike ja PR-viike kasutada loendurisse algväärtuse kirjutamiseks.



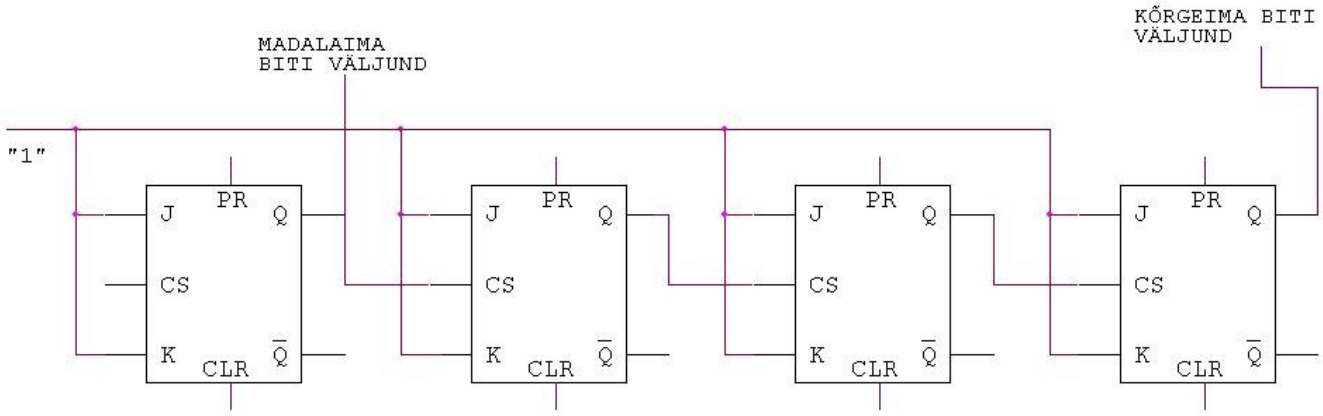
Joonis # 8



Joonis # 9



Joonis # 23

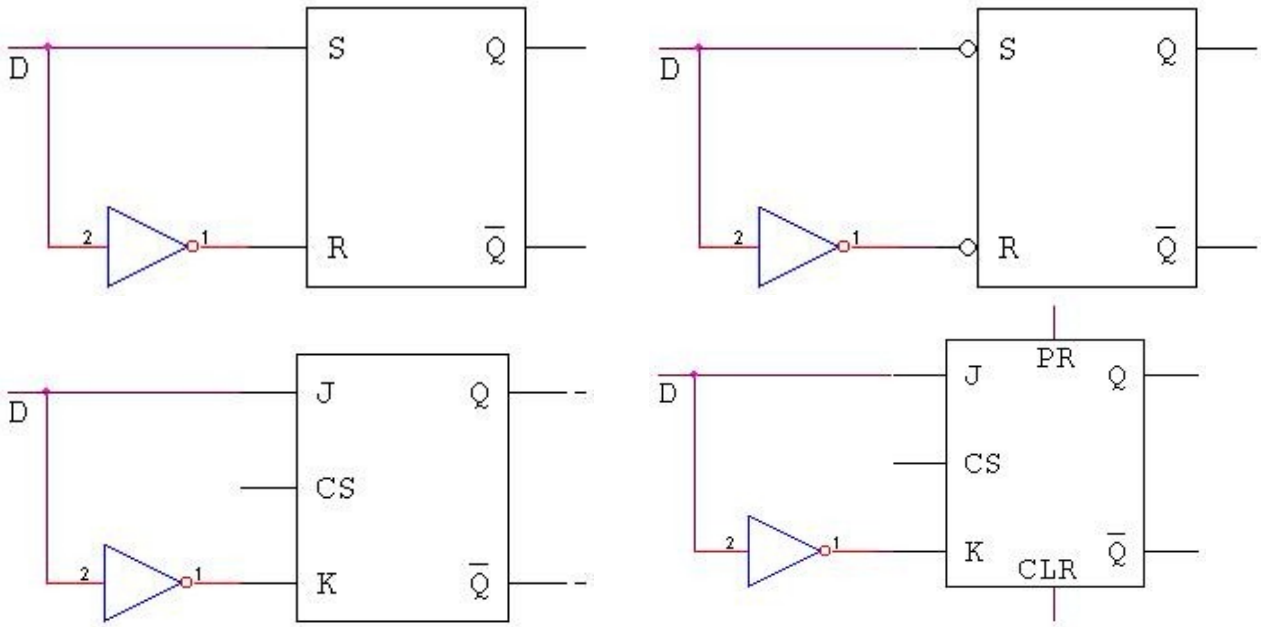


Joonis # 12

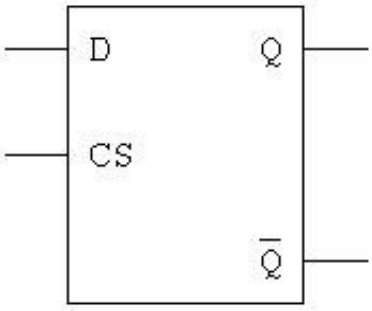
MS-tüüpi JK-trigerite kohta lisainformatsiooni saamiseks või konkreetsete aegdiagrammide vaatamiseks soovitan sisestada otsingumootorisse “Google” märksõna “DM7476” või uurida muude konkreetsete MS-tüüpi JK-trigerite andmelehti.

D-triger

Kuna D-triger(Delay Flip-Flop, Data Flip-Flop) on nii läbipaistev ja lihtne, siis joonised räägivad enda eest(joonis # 7). Pool perioodi pika viivituse annab siiski ainult MS-tüüpi JK-trigeril baseeruv D-triger.

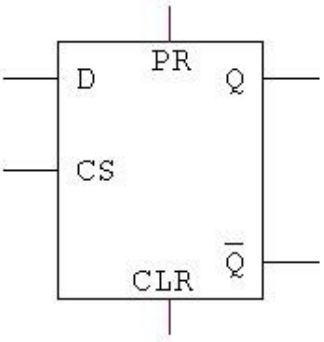


Joonis # 7



D-trigeri üldtähistus, sõltumata viivituse olemasolust.

Joonis # 13a

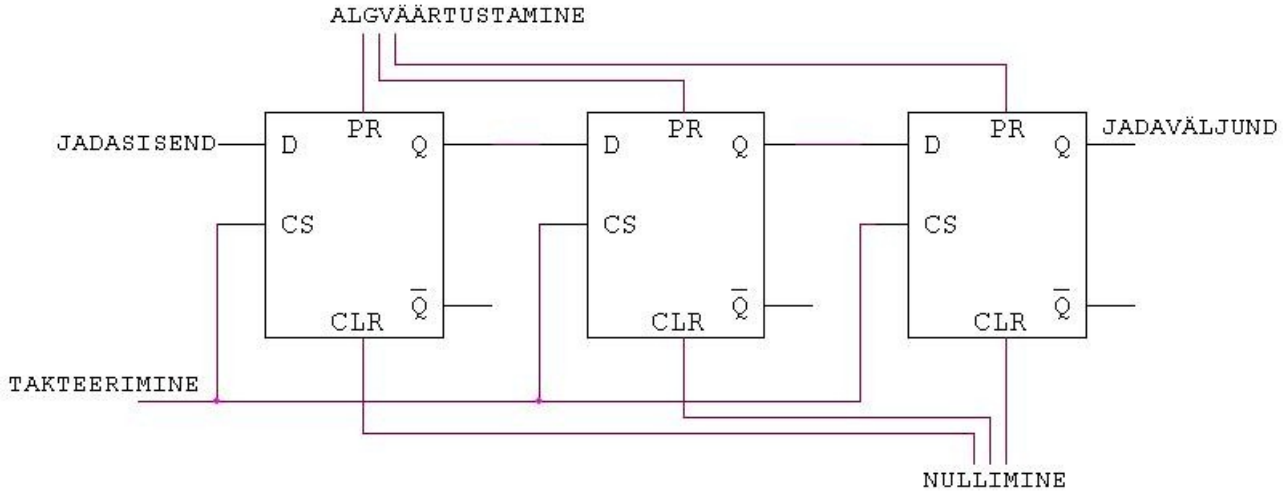


MS-tüüpi JK-trigeril baseeruva D-trigeri tähistus(viivitusega D-triger).

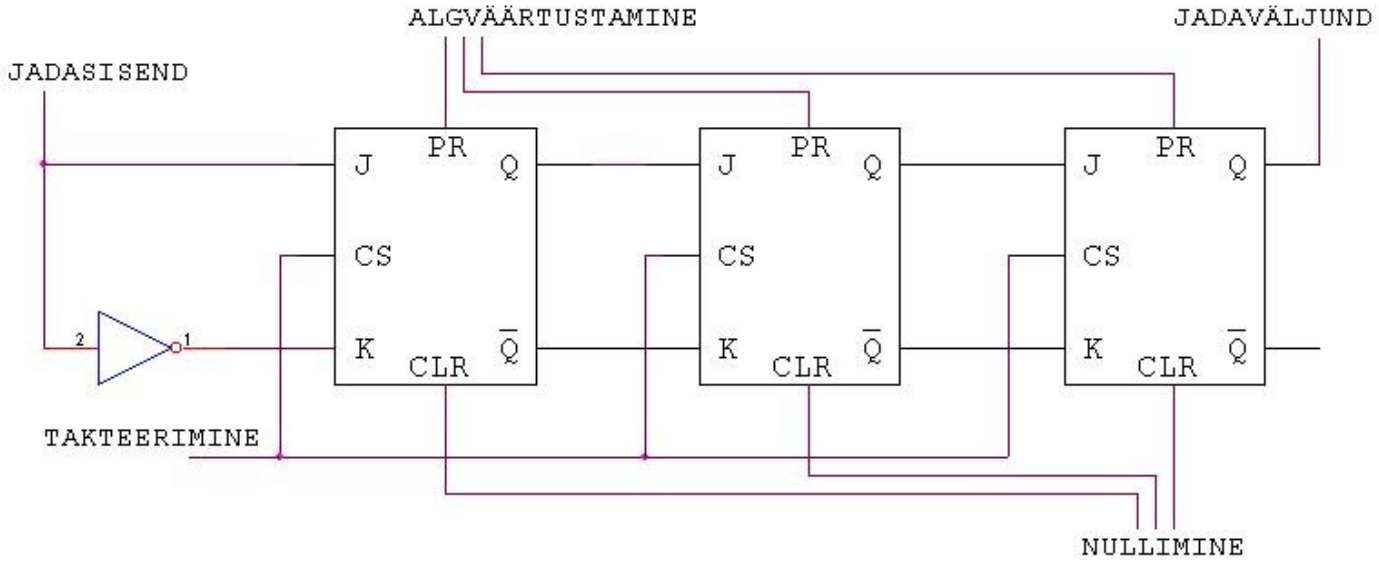
Joonis # 13b

Registreid

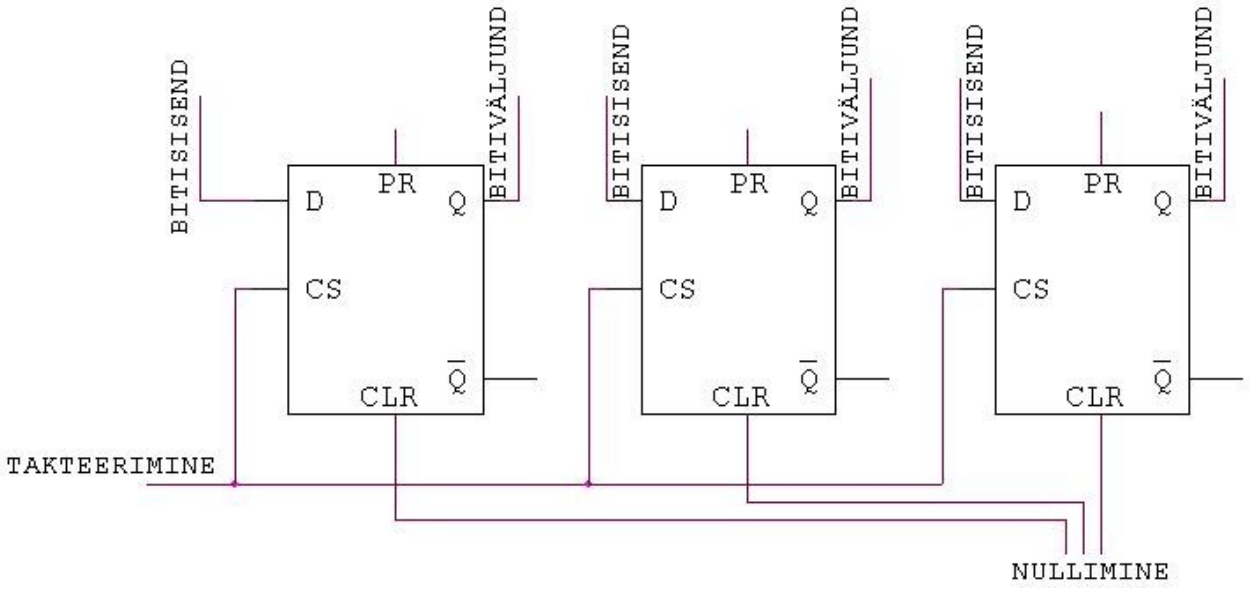
MS-tüüpi JK-trigeritest koostatud registrid on oma tööpõhimõttelt praktiliselt identsed MS-tüüpi JK-trigeril baseeruvatest D-trigeritest tehtud registritega(joonised #14 ja # 15). Nihkeregistri puhul tehakse 1 nihutamine 1 takti kohta. Nihkeregistrist osade ühenduste lahtivõtmise teel saab ülilihtsalt teha asünkroonse registri(joonis # 16). Asünkroonse registri puhul võivad D-trigerid olla ka ilma viivituseta D-trigerid, näiteks SR-trigeri baasil koostatud D-trigerid.



nihkeregister
Joonis # 14



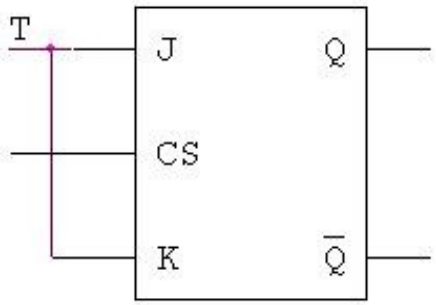
MS-tüüpi D-trigeri asemel MS-tüüpi JK-trigeri kasutamise idee seisneb selles, et kaks eitust annavad kokku jaatuse.
Joonis # 15



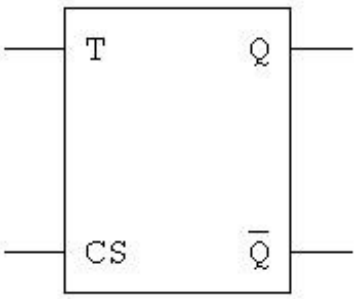
3-bitine asünkroonne register
Joonis # 16

T-triger

MITTE-MS-tüüpi JK-trigeril põhineva T-trigeri(Toggle Flip-Flop) T-viiku 1 andmisel antakse JK-trigerile keelatud sisend 11, mille peale JK-trigeri väljund on ka 11. Joonisel # 5a kujutatud trigeril on pärast T-viigu uuesti 0´llilaskmist JK-trigeri väljund juhuslik, st. et kas 10 või 01.

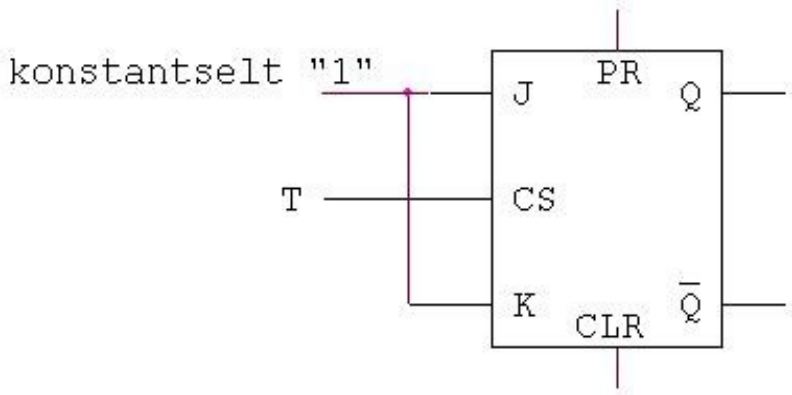


Joonis # 5a

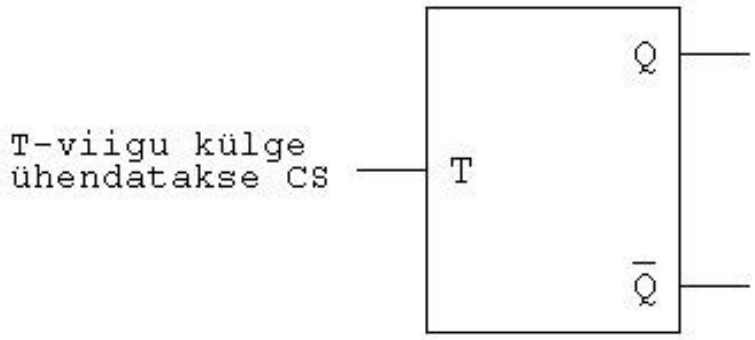


MITTE-MS-tüüpi JK-trigeril
põhineva T-trigeri tähistus
Joonis # 5b

T-trigeriks nimetaktakse mõnedes allikates ka joonisel # 5c kujutatud trigerit ning sellise MS-tüüpi JK-trigeril põhineva T-trigeri üldtähistus on joonisel #5d.



Joonis # 5c



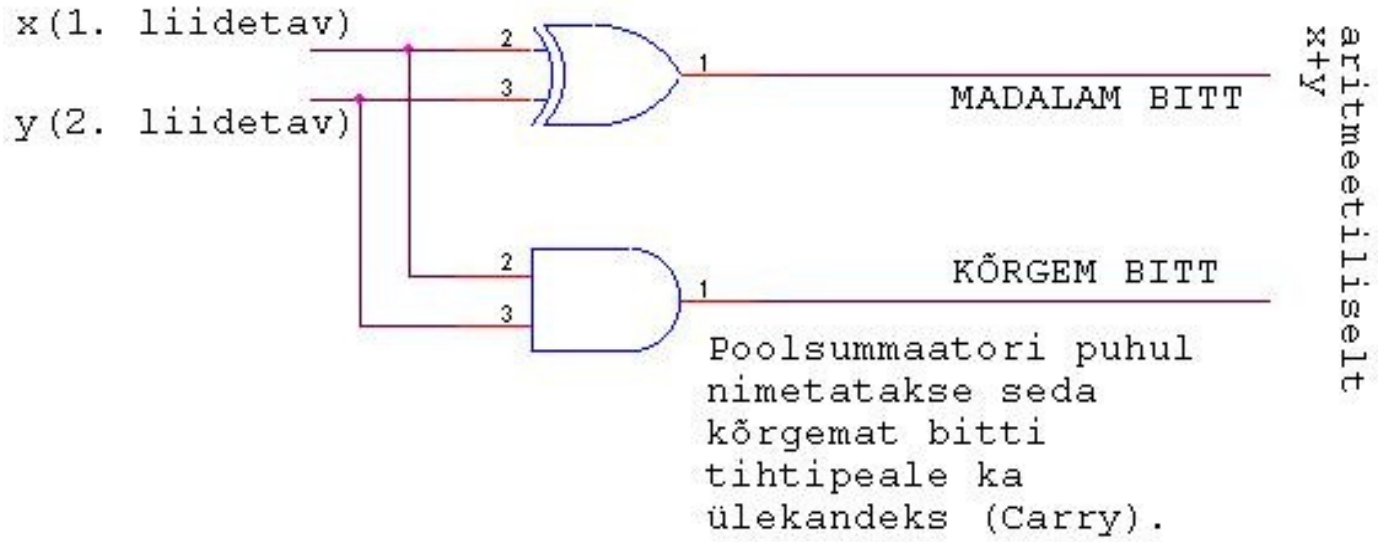
Joonis # 5d

MS-tüüpi JK-trigeril põhinev T-triger (joonis # 5d) inverteerib iga CS-taktiga oma väljundit.

SUMMAATOREID

Poolsummaator

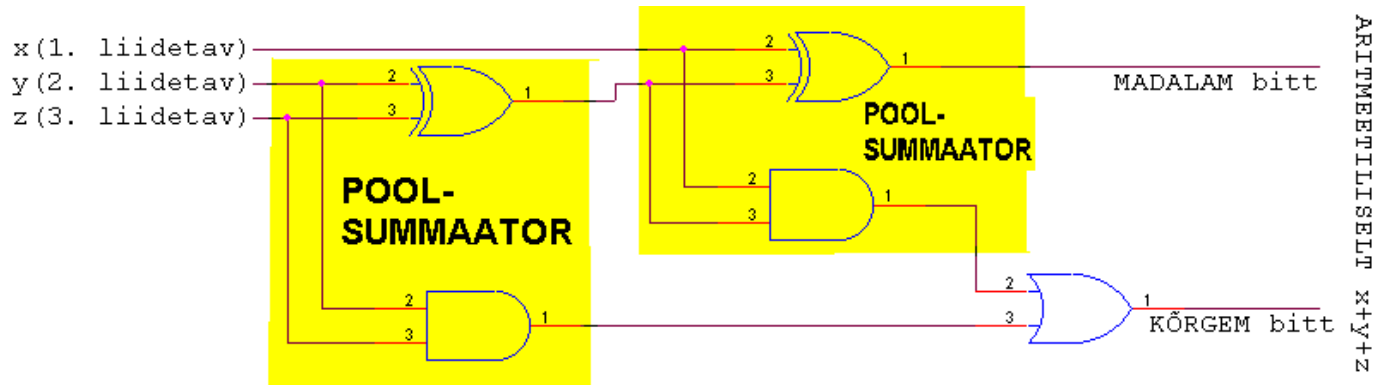
Joonis räägib enda eest(joonis # 17).



Joonis # 17

Täissummaator

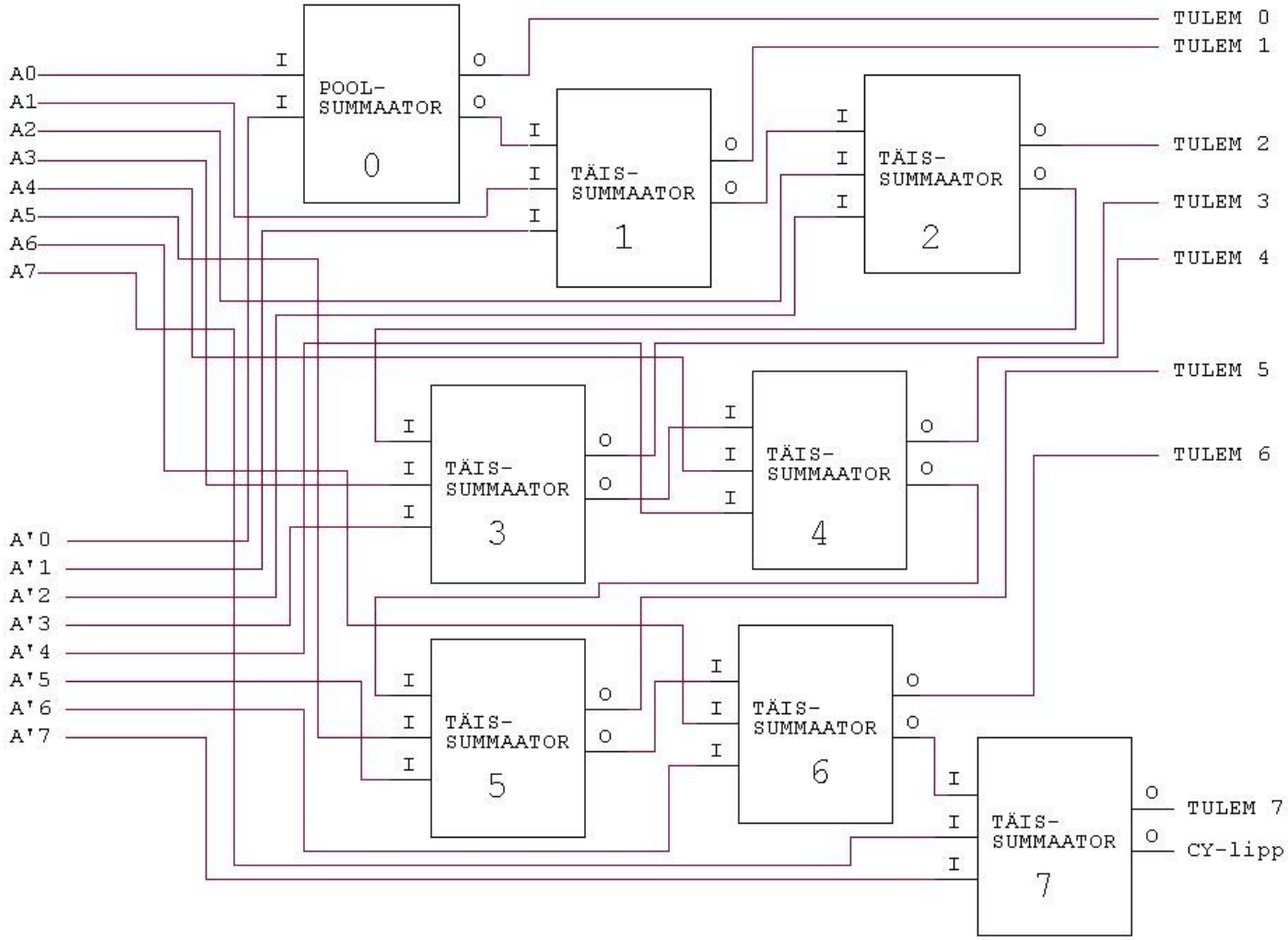
Joonis räägib enda eest(joonis # 18).



Joonis # 18

2 baidi summaator

N-positsioonilisi kahendarve kokkuliitev summaator töötab täpselt samal põhimõttel nagu paberil käsitsi kahendarve kokku liidetakse: liidan 2 numbrit kokku, kui tulemus on suurem kui 1, siis panen 1 “meelde”, järgmisel positsioonil liidan jälle 2 numbrit ja selle mis ma eelmine kord “meelde” panin kokku ja kui tulemus on suurem kui 1, siis panen jälle 1 “meelde” ja asun järgmise positsiooni numbrite kokkuliitmise kallale, jne. Joonisel # 19 on kujutatud summaator, mis liidab kokku 2 1-baidist arvu(n= siis 8).


A0
A1
A2
A3
A4
A5
A6
A7
A'0
A'1
A'2
A'3
A'4
A'5
A'6
A'7
POOL-
SUMMAATOR
0
TÄIS-
SUMMAATOR
1
TÄIS-
SUMMAATOR
2
TÄIS-
SUMMAATOR
3
TÄIS-
SUMMAATOR
4
TÄIS-
SUMMAATOR
5
TÄIS-
SUMMAATOR
6
TÄIS-
SUMMAATOR
7
I
O
TULEM 0
TULEM 1
TULEM 2
TULEM 3
TULEM 4
TULEM 5
TULEM 6
TULEM 7
CY-lipp

Joonis # 19

TUUM “ÜKSBITT”

Tuuma lühikokkuvõte

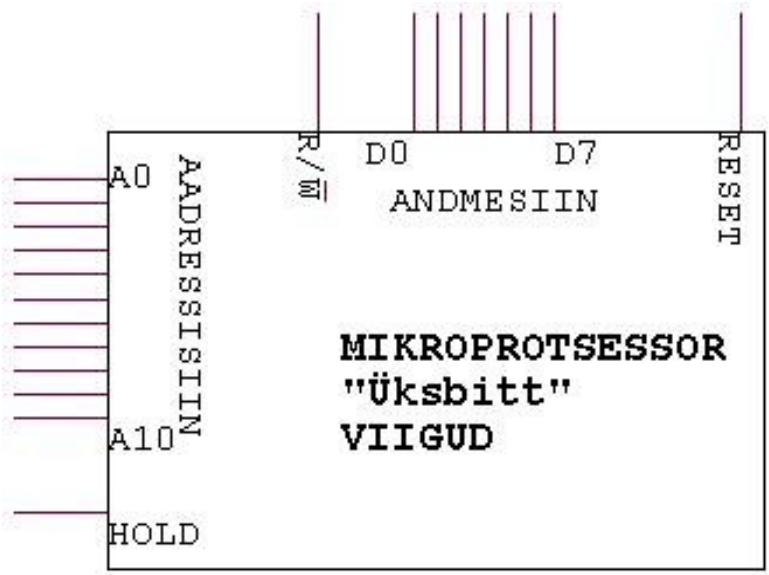
Programmeerijale nähtavad registrid:
* A----1B, akumulaator
* A'---1B, pool-akumulaator
* B----1B, saab kasutada lihtsalt panipaigaks.
* PC---11b, Aadressiloendur(Program Counter)

Võimalikud tegevused:
* Aadressilt lugemine.
* Aadressile kirjutamine.
* Akumulaatori sisu nihutamine vasakule.
* Akumulaatori sisu nihutamine paremale.
* Akumulaatori sisu inverteerimine.
* Akumulaatori ja pool-akumulaatori sisu liitmine.
* Registri sisu kopeerimine yhest registrist teise(A, A’, B).
* Registrisse konstandi sisestamine(A, A’, B).
* Tingimusteta siirdumine.
  A’ sisu kopeeritakse PC kõrgematesse bittidesse ja B sisu kopeeritakse PC madalamasse baiti.
* Hargnemine kui Z=="1"

Programmeerijale nähtamatud ehk süsteemsed registrid:
* K----2B, käsudekodeerimisregister, kus asub hetkel täidetav käsk
* K'---2B, "instruction fetch" ehk käsulugemisregister,
  kuhu laetakse järgmisena täitmisele kuuluv käsk.
* D----Andmesiiniregister
* Ahl---11b, Aadressisiini register
*n pA----1B, n preAkumulaatorit ehk eelakumulaatorit, kuhu
  pannakse vahetu tehte tulemus enne selle akumulaatorisse
  panemist. See, millise tehte tulemus akumulaatorisse laetakse,
  määratakse käsudekoodri poolt.
* pPC--- 11b, prePC, praktiliselt sama otstarbega nagu preAkumulaator, ainult et
  aadressiloenduriga opereerimiseks.

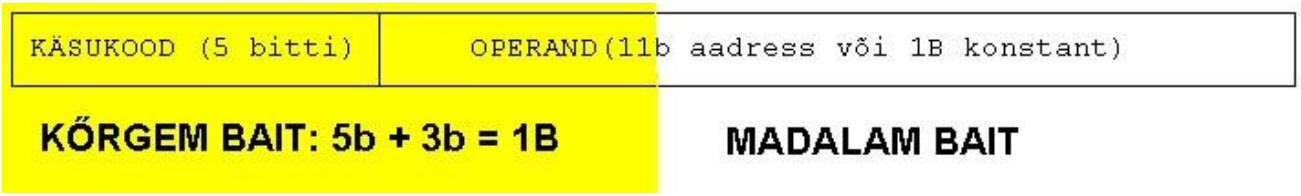
Protsessori Lipud:
- Cy---1b, “Carry” ehk ülekandelipp
- Z----1b, “Zero” ehk nulltulemuse lipp

Järgnevalt on loetletud MITTE-MS-tüüpi JK-trigeritel põhinevatest D-trigeritest tehtud asünkroonseid registreid: A, A’,B,K, K’,Cy, Z, pA(n tükki). Nihkeregistrite puhul eeldan, et nad on koostatud joonisel # 23 näidatud MS-tüüpi JK-trigeritest.



Joonis # 38

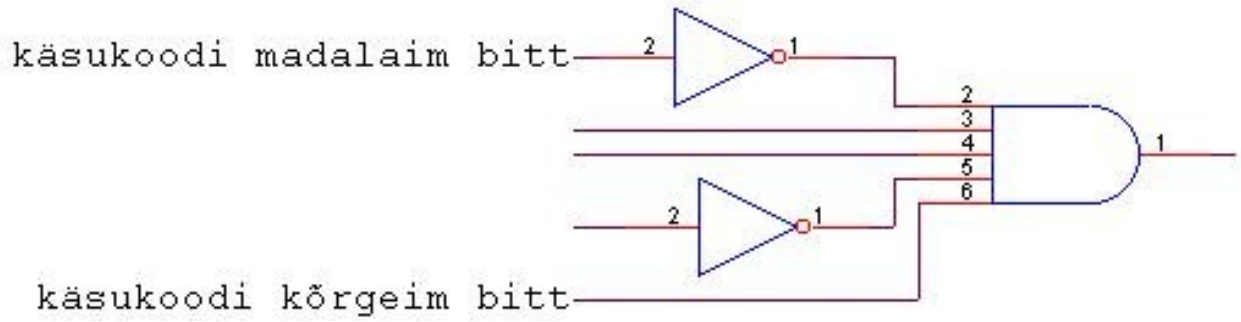
Käsustik



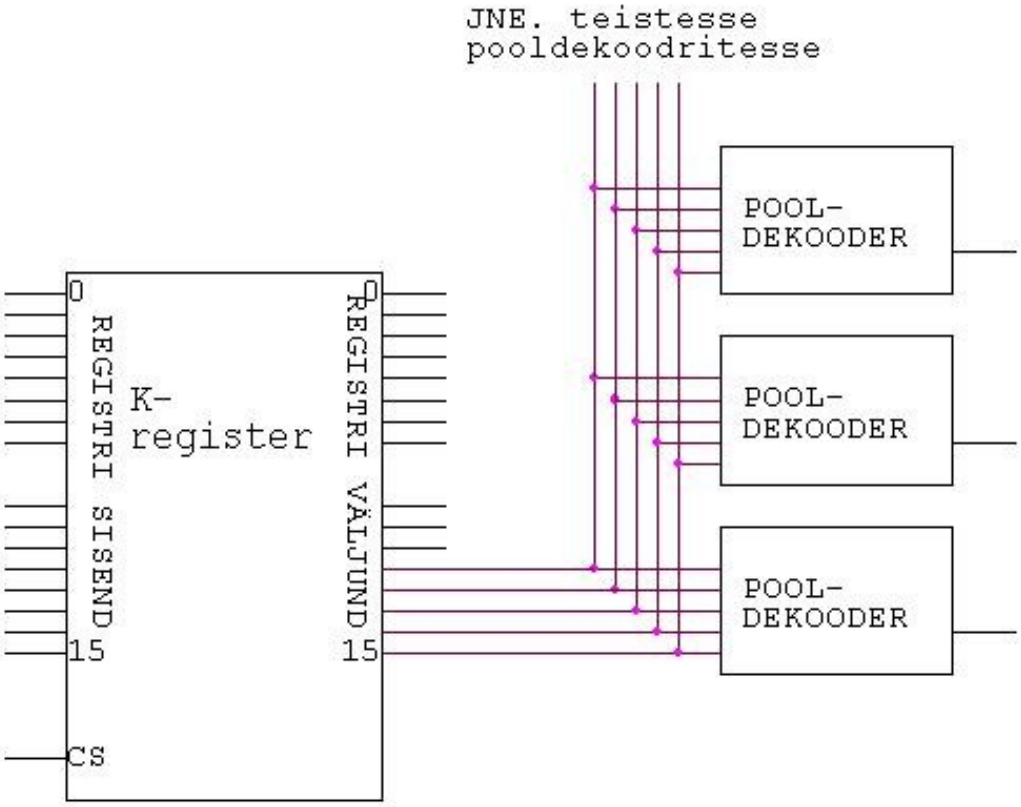
Joonis # 24

Erinevaid käsukoode on kokku 18, sellest 6 tükki moodustavad sisukopeerimiskäsud registrite A, A’ ja B vahel ning 3 tükki on registritesse A, A’, B konstandi sisestamiseks. Kõik käsud on ühepikkused ning nagu jooniselt #24 juba järeldada võib, on aadressiruumi suuruseks $2^{11}B = 2KB$.

Käsudekooder kasutab iga käsukoodi jaoks eraldi eitustest ja konjuktsioonidest kokkupandud lülitusi, mis ühendatakse kõik paralleelselt registri K kõrgema baidi kõrgeimate bittidega. Näide käsukoodi 10110 jaoks on toodud joonisel # 25. Nimetan omavoliliselt neid lülitusi “pooldekoodriteks”.

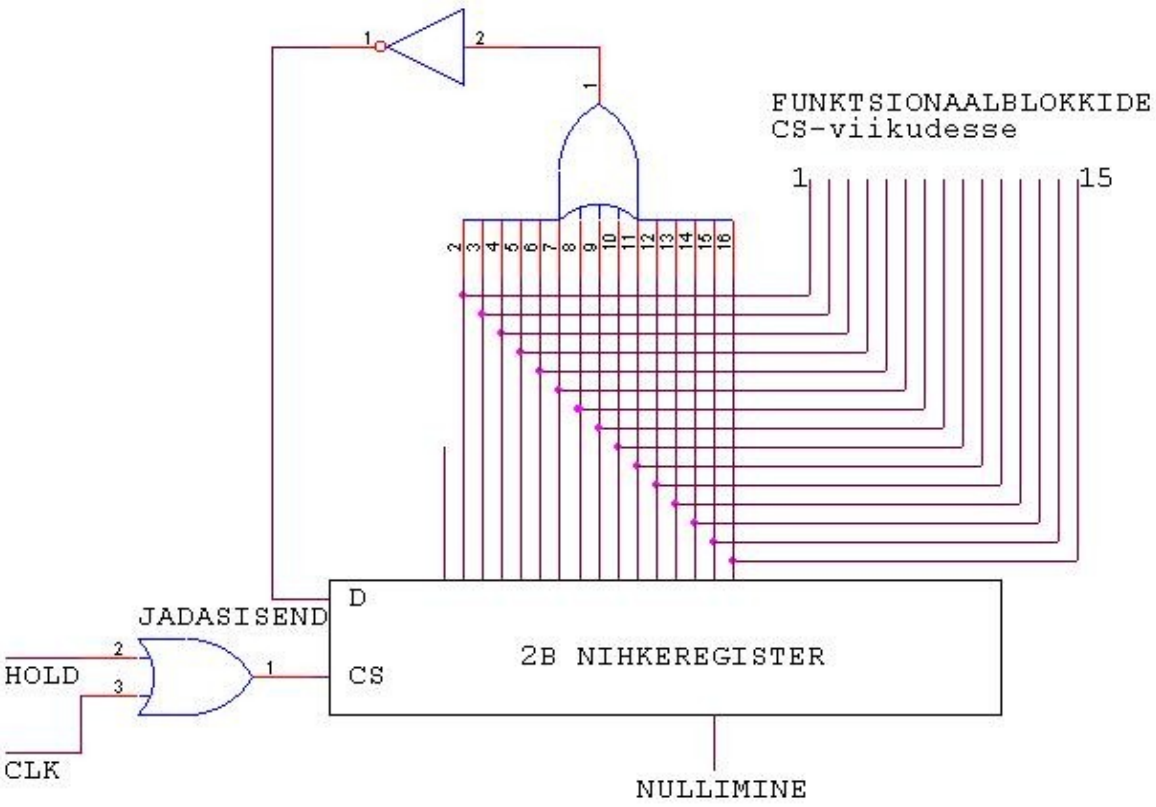


Joonis # 25



käsu dekodeerimine
Joonis # 26

Masinatsükkleid genereeritakse nihkeregistriga, kus automaatselt nihkub üksik “1” vasakult paremale (joonis # 28).



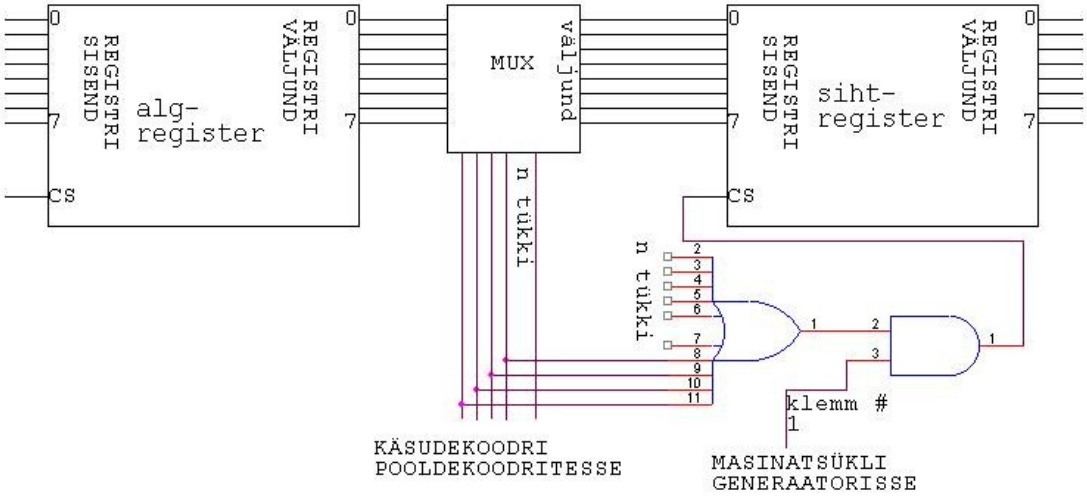
masinatsükli generaator
Joonis # 28

Funktsionaalblokkides kasutan ma multipleksoreid, mis on oma olemuselt nagu joonisel # 27 kujutatud multipleksor, kuid ilma dekoodrita ja väljundkanalite ning sisendkanalite arvud on erinevad.

Tegelikult on peaaegu kõigi registrite CS-sisendi ees VÕI-lülitus, kuid paraku unustasin ma hajameelsusest need joonistele märkida.

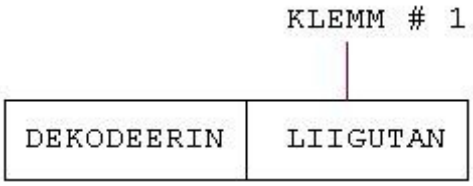
Ühe registri sisu teise kopeerimine

MOV-käsk on realiseeritud joonsiel # 29 kujutatud skeemi abil.



Joonis # 29

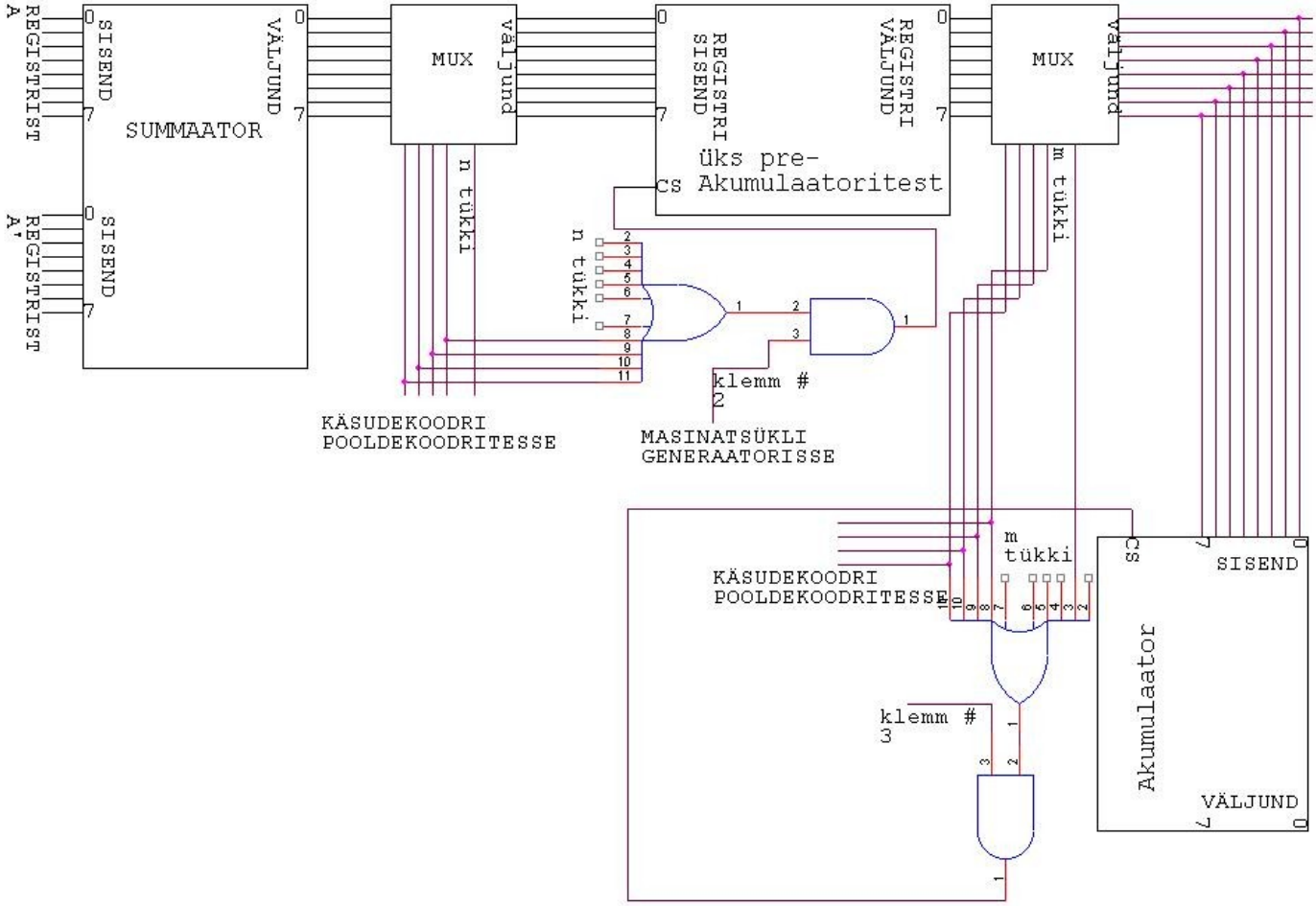
Sama skeemi abil realiseerin ka konstandi kirjutamise registrisse. Konstandi registrisse kirjutamise puhul on algregistriks lihtsalt käsuregistri K madalam bait. Kopeerimise masinatsükli ajadiagramm on kujutatud joonisel # 30.



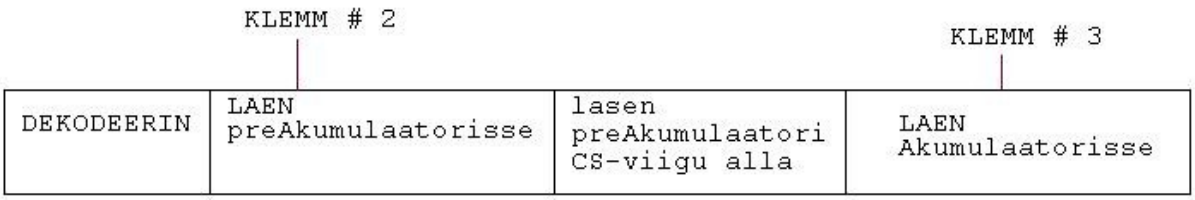
Joonis # 30

Summeerimine

Joonisel # 31 olev preAkumulaator on vajalik seoses sellega, et Akumulaator kui tulemuse sihtpunkt sisaldab ka ühte liidetavat.



Joonis # 31

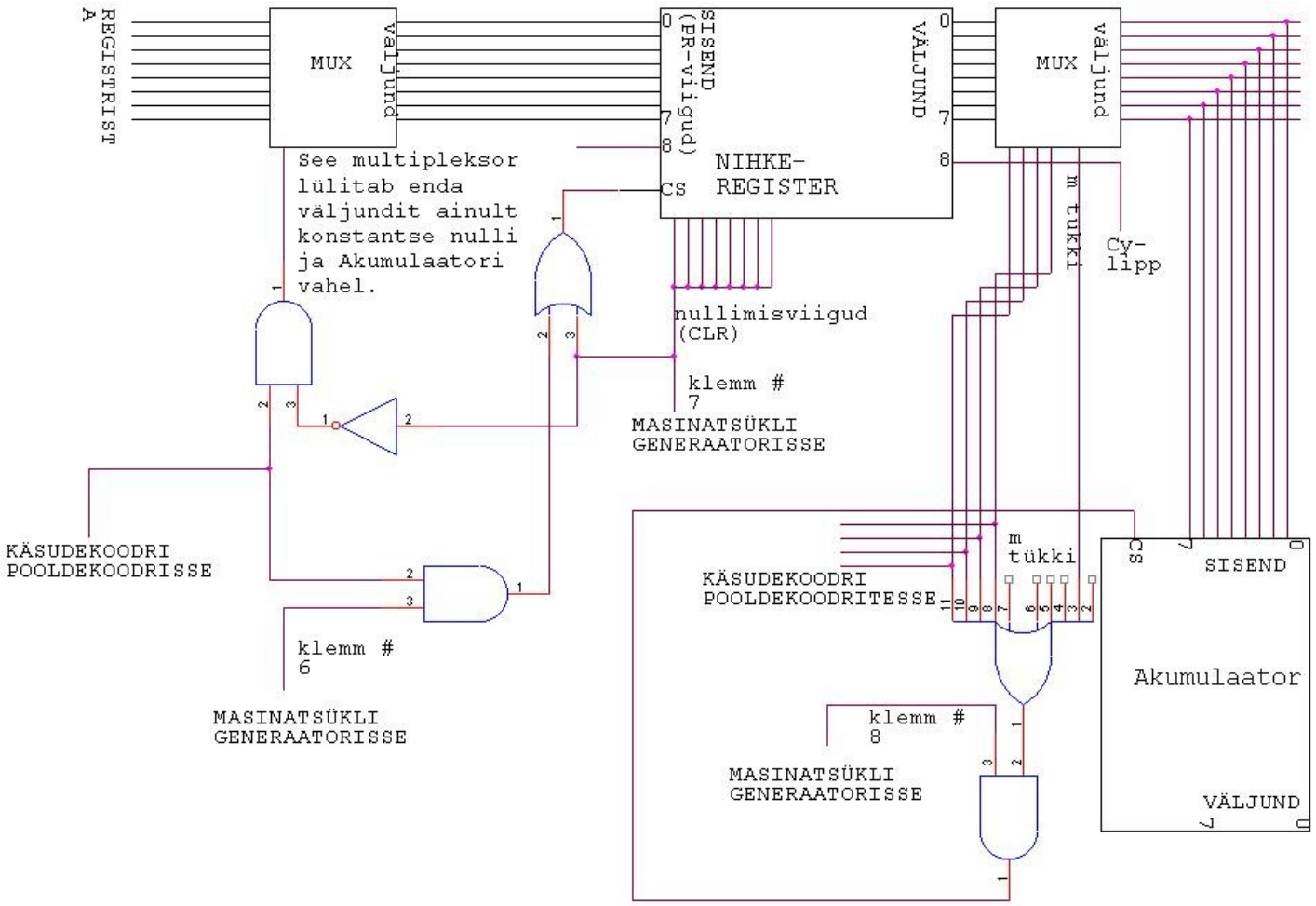


Joonis # 32

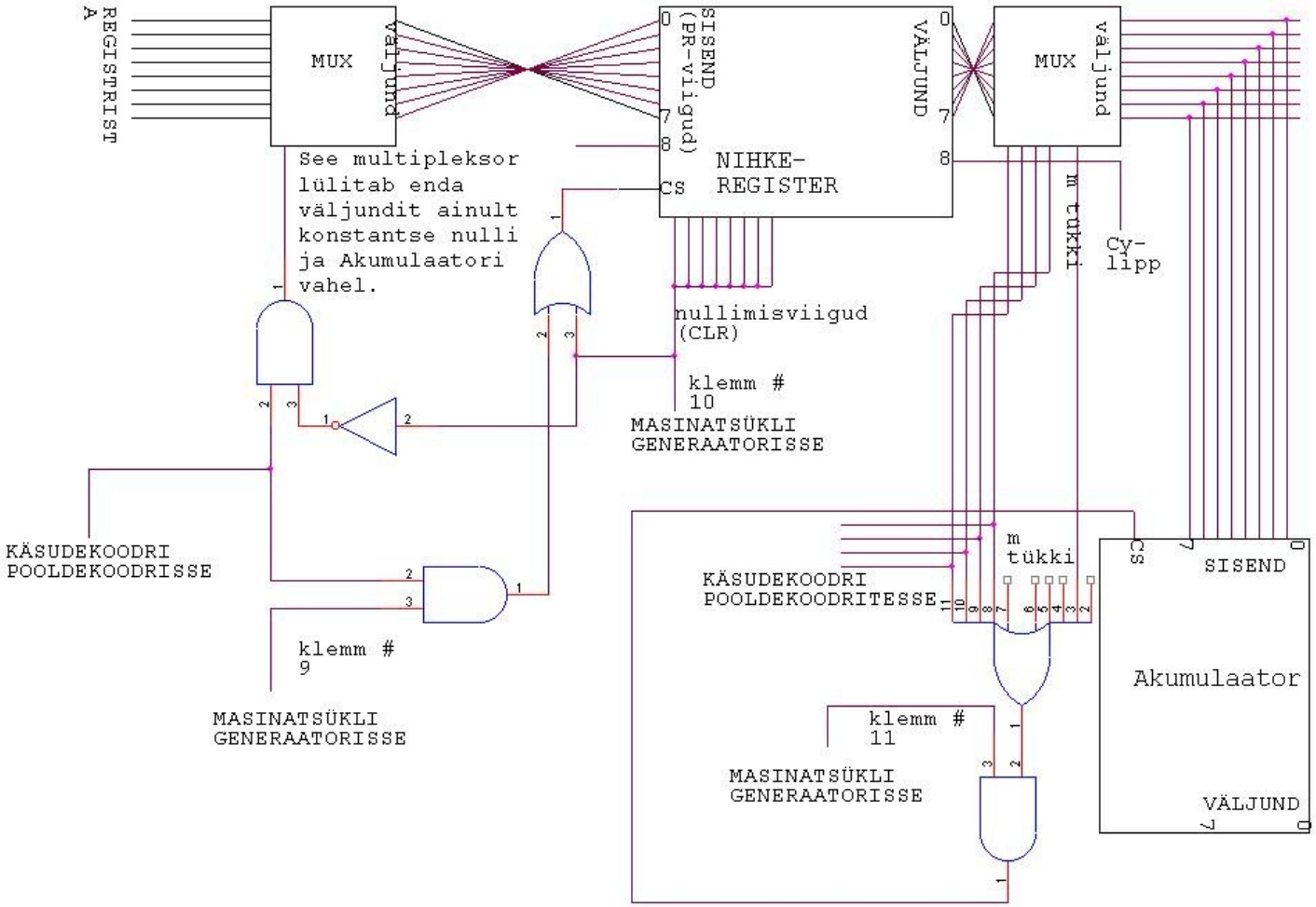
Inverteerimine

Inverterimise puhul kasutan täpselt samasugust skeemi nagu on summeerimise puhul(joonis # 31), ainult et klemm # 2 on asendatud klemm # 4, klemm # 3 on asendatud klemm # 5, summaator on asendatud inverteriga, mille sisend on ühendatud preAkumulaatoriga. Loomulikult on ühilduvad ka summeerimise ja inverteerimise aegdiagrammid(joonis # 32).

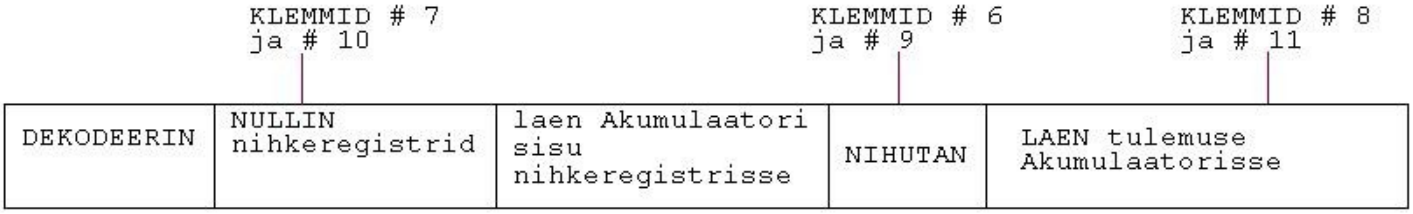
Nihutamine



nihutamine vasakule
Joonis # 33



nihutamine paremale
Joonis # 34

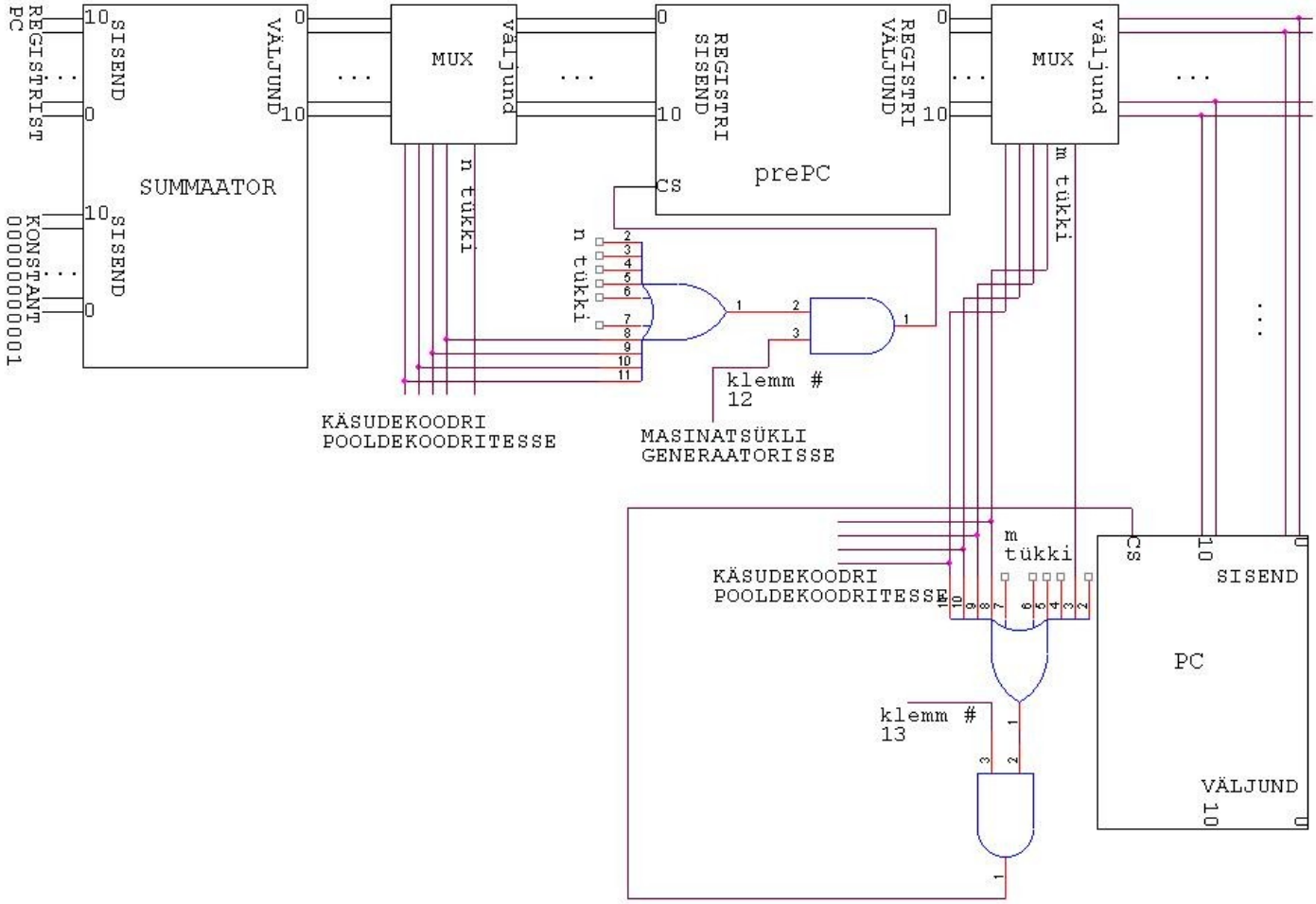


nihutamise ajadiagramm
Joonis # 35

Siirdumine

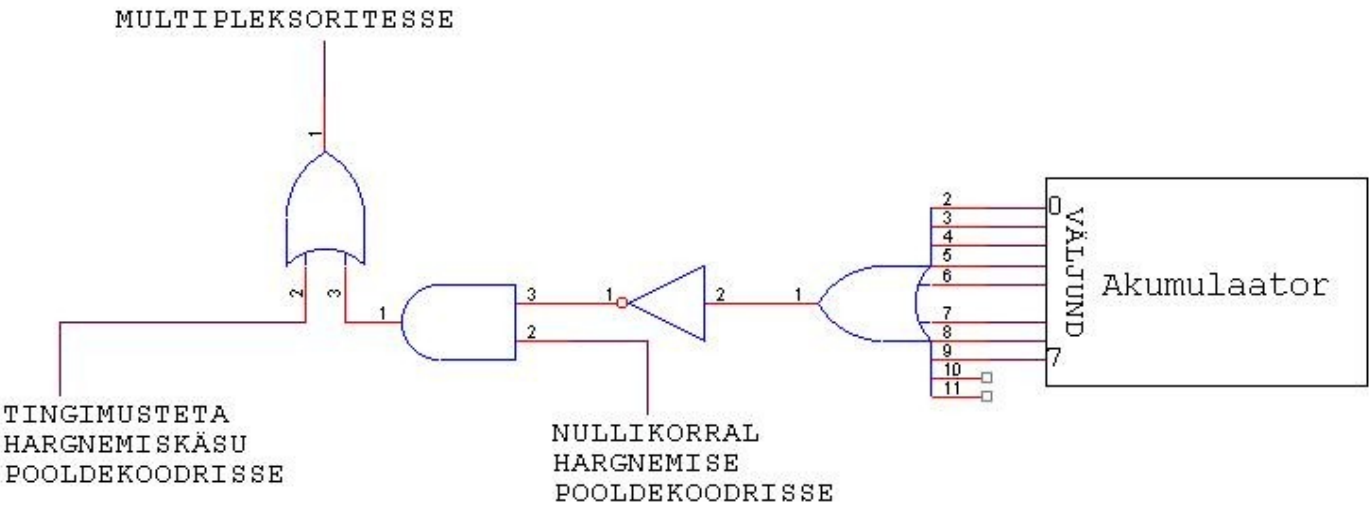
Siirdumine järgmisele käsule käib loomulikult automaatselt. Sellisel juhtumil registrit PC lihtsalt inkrementeeritakse(Joonis # 36), RW-jalg on vaikimisi “1”, kopeeritakse analoogiliselt MOV-käsu täitmisele(joonis #29) PC sisu aadressisiiniregistrisse, oodatakse 1 masinatsükli takt, siis kopeeritakse, ka analoogselt MOV-käsu täitmisele, ainult et algregistri asemel on siin, siini väärtus K-registri madalamasse baiti. Seejärel loetakse sama moodi sisse K-registri kõrgem bait.

Juhul kui on tegu programse tingimusteta siirdumisega, siis toimitakse samamoodi nagu automaatse siirdumise puhul, ainult et esimese PC-inkrementeerimise asemel kopeeritakse B-registri sisu PC madalamasse baiti ning registri A’ madalamad bitid PC kõrgematesse kolme bitti.



PC inkrementeerimise ajadiagramm on ühilduv summeerimise ajadiagrammiga.
Joonis # 36

Juhul kui hargnemist teostatakse tingimusel, et Akumulaatori sisu on null, siis saab kasutada tingimusteta hargnemise funktisionaalblokke ja masinatsüklit. Täpsemalt: joonis # 37.



Joonis # 37

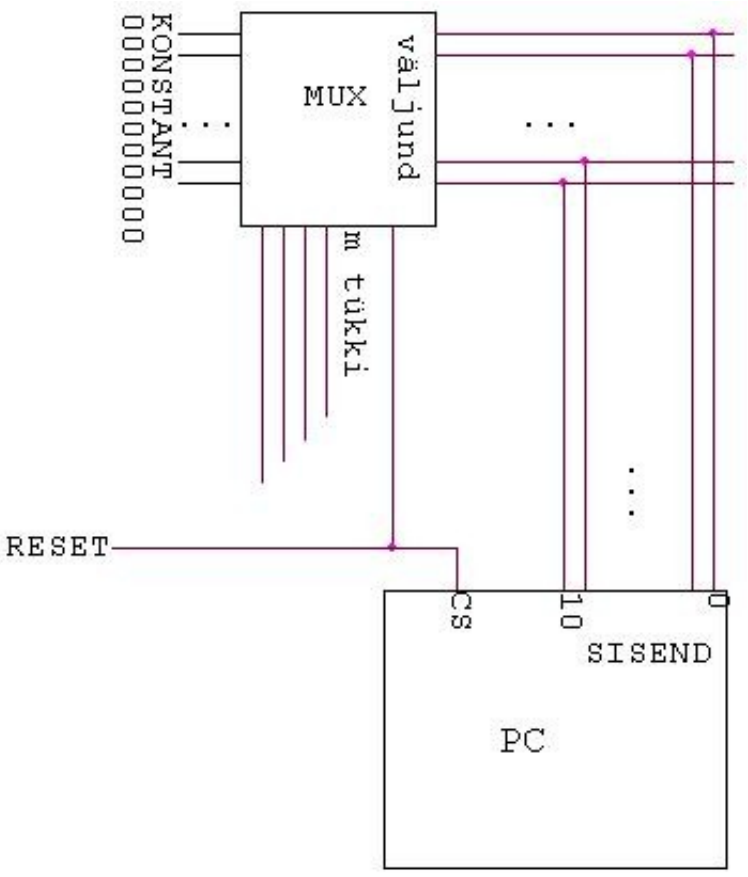
Andmete lugemine, kirjutamine

Kuna RW-jala väärtus on vaikimisi kõrge, siis seda lugemise korral eraldi seada ei ole vaja. Kõigepealt kopeeritakse K-registri madalamad 11 bitti aadressisiiniregistrisse, oodatakse 2 masinatsükli takt ja siis kopeeritakse andmesiinil olev Akumulaatorisse. Vahepeal on mälul või välisseadmel võimalik seada HOLD-viik kõrgeks, mille tulemusena masinatsüklite genereerimises tekib viivitus(joonis # 28).

Kirjutamise korral laetakse kõigepealt K-registri 11 madalamat bitti aadressisiiniregistrisse, seejärel kopeeritakse Akumulaatoris olev andmesiiniregistrisse, siis seatakse RW-jala väärtus madalaks, “0”, oodatakse 2 masinatsüklit, seatakse RW-jala väärtus tagasi kõrgeks, “1”.

Reset

Reseti puhul laetakse PC-sse konstantne 0, nagu ikka(joonis # 39).



Joonis # 39

KASUTATUD MATERJALID

- Dr. Ando Ots’a loengutes jagatud materjalid ja minu enda konspektid
- Dr. Matti Fischer’i loengutes jagatud materjalid ja minu enda konspektid
- “Digital Principles”, Roger L. Tokheim, McGRAW-HILL BOOK COMPANY, 1980
- Erinevaid materjale Internetist