Üle-eelmise näitena toodud intervalli vektoresitus on **0 — — :**
**{ 000 001 010 011 } = 0 — —**

Eelmise näiteintervalli vektoresitus on **0 1 — 0 :**
**{ 0100 0110 } = 0 1 — 0**

## LOOGIKAELEMENDID — digitaalsignaalide töötlejad

Kahendkoode (ehk nende koosseisu kuuluvaid loogikaväärtusi **0** **1** ) töötlevat elektriskeemi nimetatakse **digitaalskeemiks**.

Iga digitaalseadme elementaarseteks koostisosadeks on **loogikaelemendid**, mis teevad loogikaväärtustega **0** ja **1** lihtsaimaid loogikatehteid — ehk töötlevad "*ühtesid*" ja "*nulle*" (uuteks) "*ühtedeks*" ja "*nullideks*".

*Loogikaelementide* omavahelisel kokkuühendamisel saadakse **loogikaskeem / loogikalülitus / digitaalskeem / digitaalseade** .

Iga digitaalseade koosneb seega loogikaelementidest ja ta töötleb **1**-de ja **0**-de kogumeid. Digitaallülituse igal signaaliliinil, sisendil, väljundil (igal "traadil") on igal hetkel olemas kumbki alternatiivne väärtus 0 või 1, mis välistavad teineteist.

*püstitame küsimuse:*

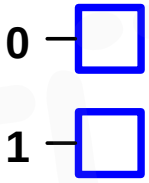
**?** Millised on **lihtsaimad võimalikud** (elementaarseimad) digitaalsignaalide töötluselemendid (loogikaelemendid), millest saaks koostada suuremaid / keerukamaid digitaalseadmeid **?**

Vaatleme esmalt väga lihtsaid "digitaalkomponente" , millel on:

— **1 sisend** ja **mitte ühtegi** väljundit;

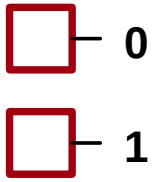
— **1 väljund** ja **mitte ühtegi** sisendit;

**ainult sisend** olemas :




kasutu, kuna "töötulemust" ei väljastu

**ainult väljund** olemas :

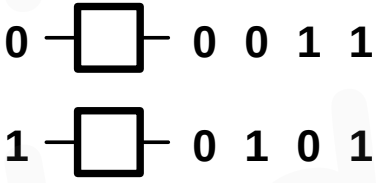



*konstandi generaatorid* (kuid ikkagi nad ei ole **töötlevad** elemendid)

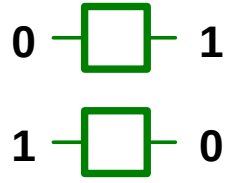
Sellised elemendid ei osutu digitaalsignaali lihtsateks töötlejateks — seega nad ei paku meile huvi, kuna oleme otsimas digitaalväärtusi (1 ja 0) **töötlevaid** lihtsaimaid võimalikke komponente/seadmeid.

Läheme sammuvõrra keerulisemaks: vaatleme seadet millel on **1 sisend** ja **1 väljund**.

Võtame seda seadet **2 eksemplari** (vaatleme sama seadme kahte identset koopiat):




**4 erinevat võimalikku reaktsiooni**




kasulik töötlustulemus

Seega eksisteerib täpselt **4** erinevat võimalikku seadet/elementi, millel on *üks sisend* ja *üks väljund*.

**neljast** erinevast võimalikust reageerimisviisist osutub **töötlevaks** reageerimiseks ainult üks — **inverteerimine**;

ülejäänud 3 on "sisendsignaali (loogiliselt) mittetöötlevad" reageerimisviisid:

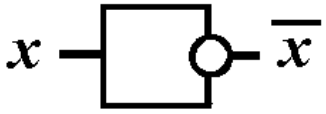
— *konstant* **0** *generaator*

— *konstant* **1** *generaator*

— *sisendsignaali taaskordaja* ehk *repeater*

oleme saanud esimese kasuliku, lihtsaima digitaaltöötluselemendi:

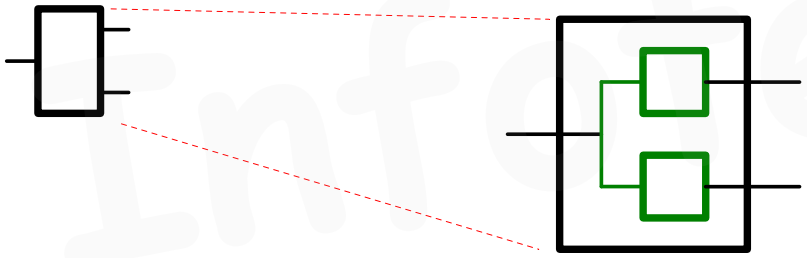
**INVERTORi** tingmärgid loogikaskeemides


Joonisena esitatud loogikaskeemides kasutatakse loogikaelementide tähistamiseks spetsiaalseid *tingmärke*.

Lihtsaim loogikaelement on **invertor** ehk **EI**-element ( **NOT** ).
*Invertor* teostab loogikamuutuja inversioonitehet ehk eitust.

Jätkame muude võimalike (lihtsate) töötluselementide otsimist.
Läheme jälle sammuvõrra keerulisemaks: lisame juurde ühe **väljundi** , misjuhul on elemendil **1 sisend** ja **2 väljundit** :



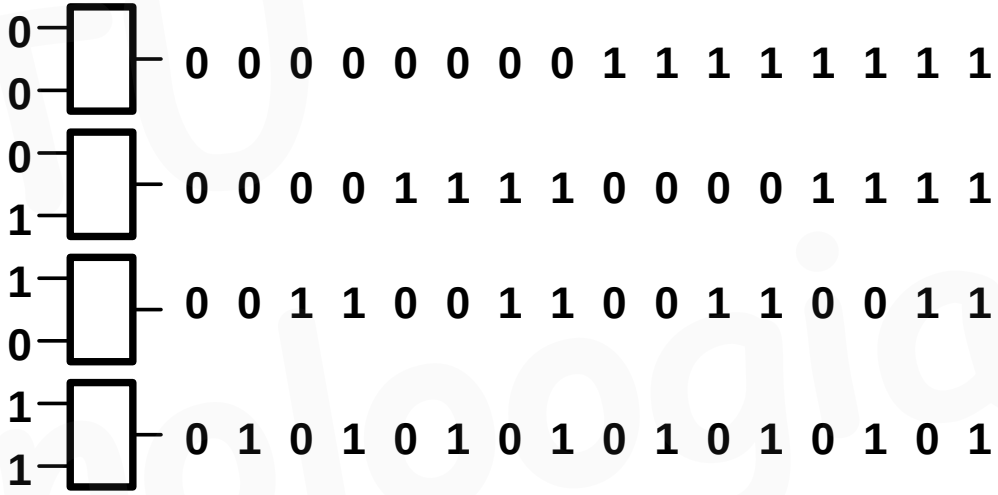
selline element osutub koosnevaks **kahest** lihtsamast elemendist

Kuna digitaalelement **1-sisend-2-väljundit** on alati koostatav kahest lihtsamast: **1-sisend-1-väljund** elemendist (*konstandigeneraatoritest* ja *invertoritest*), siis selline **2 väljundiga** element ise ei kuulu olemuselt "lihtsaimate" hulka ja seega ta meile huvi ei paku.

Modifitseerime eelmist elementi: olgu tal **2 sisendit** ja **1 väljund**

Kuigi selline element omab samuti kokku 3 sisendit-väljundit, siis tema käitumine võib olla eelmisest (1-sisend-2-väljundit) elemendist oluliselt erinev, nii et see ei ole enam koostatav *1-sisend-1väljund* elementide abil (ehk *konstandigeneraatorite 0/1* ja *invertorite* abil)

Vaatleme sellise elemendi **nelja** eksemplari
(4 identset koopiat samast digitaaltöötluskomponendist *2-sisendit-1väljund* ):



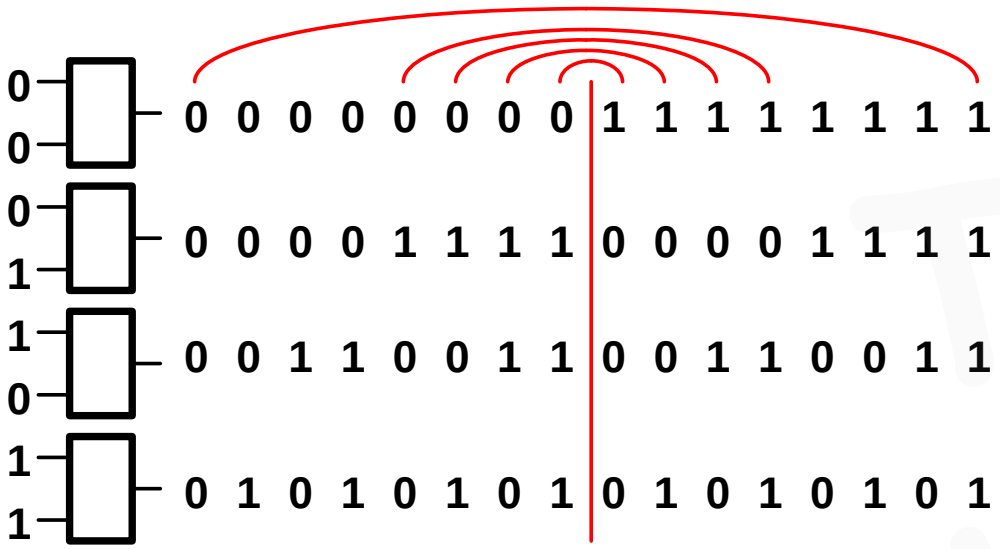

**2sisendiga1väljundiga digitaaltöötluselemendi kõikvõimalikud 16 reageerimisviisi**

ilmneb, et *2-sisendiga-1-väljundiga* töötluselement saab sisendmõjutustele reageerida ehk "käituda" **16nel erineval viisil** — ehk üldse eksisteerib **16 erinevat** võimalikku 2 sisendiga loogikaelementi.

**Millised nendest 16st eksisteerivast elemendist on oma "käitumiselt"** (ehk oma töötlustoimingu iseloomult) **kõige lihtsamad ?**

**?**

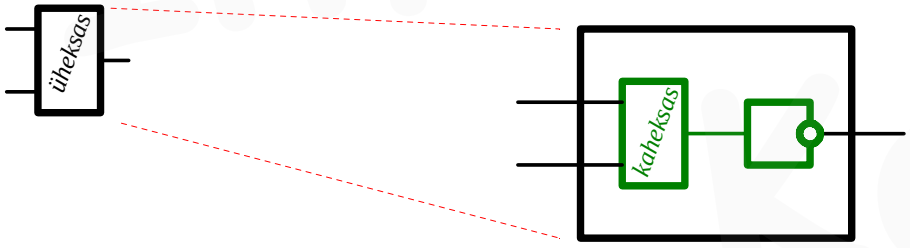
analüüsime eelnevalt väljakirjutatud kõikvõimalike reaktsioonide jada (16):




viimased 8 on esimese 8 **inversioonid**

väljundreaktsioonide esimene ja teine pool on sümmeetriliselt **teineteise inversioonid** (mõttelise "poolitaja" suhtes)

Viimast 8 väljundreaktsiooni realiseerivad loogikaelemendid oleks seega saadavad esimest 8-t realiseerivatest elementidest, kui nende väljund suunata läbi invertori:




(näiteks) **9.** element on koostatav **8.** elemendist + **invertorist**

Seega viimased 8 väljundreaktsiooni (16st) võib jätta kõrvale, kui me jätkuvalt otsime **kõige lihtsamaid** loogilise töötluse viise.

(viimased 8 on esimesest kaheksast kergesti saavutatavad / tuletatavad inversiooni kaasabil).

Jätkame **esimese 8** võimaliku väljundreaktsiooni analüüsimist.

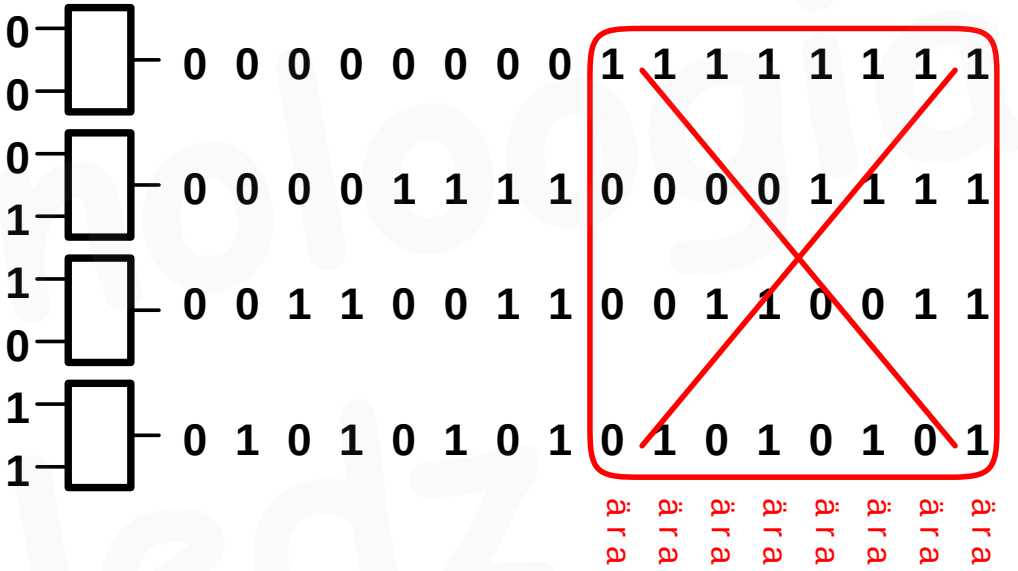
aga kas võiks "ärajätta" hoopis esimesed 8 ja jätkata lihtsate töötlusviiside otsimist "tagumise" 8 hulgast **?**

esimesed 8 võimalikku väljundreaktsiooni on ju samuti saadavad viimastest 8st, sarnasel kombel ehk **inverteerimise** kaudu **?**

**?**

**EI !** viimased 8 võimalikku "käitumist" (väljundreaktsioonide kombinatsiooni) on oma sisult kindlasti "keerulisemad" kui esimesed 8. (*miks ?*)

Seega jätame alles esimesed 8 ja jätame kõrvale viimased 8.

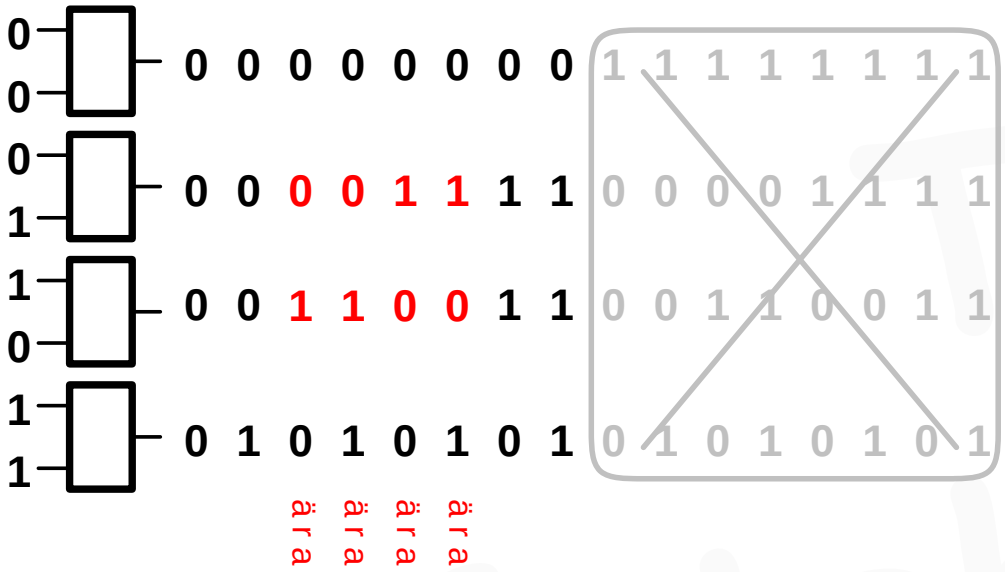



Jätkame ja võtame juurde järgmise kriteeriumi:

Sisendsignaalide paari lihtne töötlusviis peaks olema selline, mis ei sõltu **sisendite järjekorrast**: ehk sisendväärtuste paar **0 1** ja **1 0** peaks olema (töötluse mõttes) samaväärsed ehk loogilise töötluse tulemus **peaks olenema** ainult sisendite väärtustest 1 / 0 — mitte signaalide sisenemise **järjekorrast** töötluselementi.

(*võrdle aritmeetikatehete kommutatiivsusega — aritmeetiline liitmine ja korrutamine samuti ei olene operandide järjekorrast* )

Seega järgmisena eemaldame esimesest 8 veerust need veerud, kus **keskmised** read ehk **2**. ja **3**. rida on erinevate väljundreaktsioonidega:

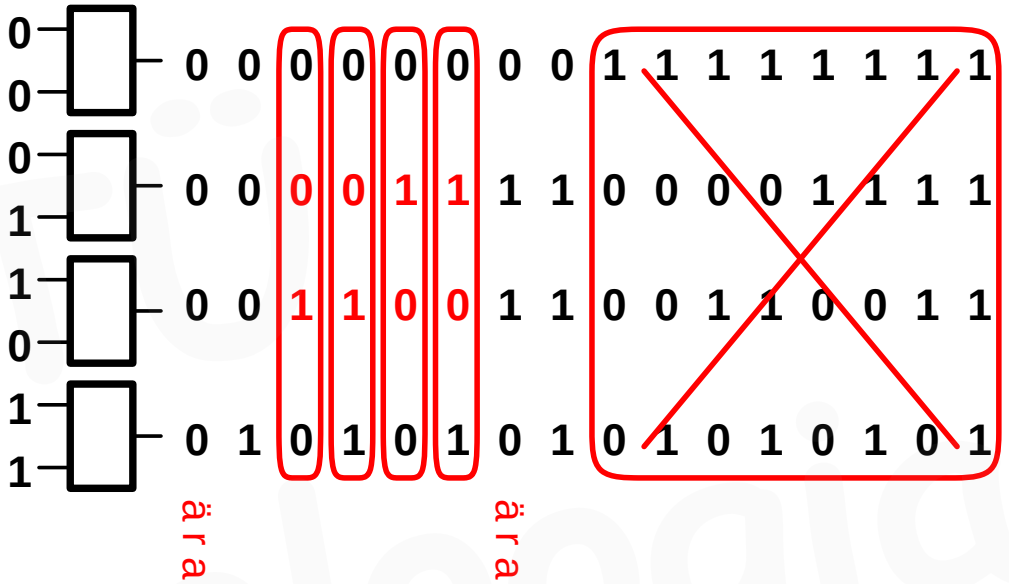



*edasi:*

**esimene** väljundreaktsioonide kombinatsioon on **konstant 0 generaator** see ei ole sisendsignaalide (loogiline) töötlus ja jätame sellise seadme ära;

**seitsmes** väljundreaktsioonide kombinatsioon on elemendi selline käitumine, kus sisendpaari **0 0** korral **ei toimu inversiooni** kuid sisendpaari **1 1** korral **toimub** sisendväärtuste **inversioon**.

**7.** veerg (käitumine) ei esita seega olemuselt lihtsat töötlust — läheb ära.




Niisiis on 16st veerust (digisignaalide paari kõikvõimalikest töötluskombinatsioonidest) alles jäänud ainult **2** veergu (2 elementi):

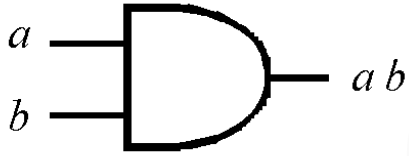

Oleme saanud **kõige lihtsamad** 2 operandiga (ehk *binaarsed* ) **loogikatehted**. Varem tuletatud *eitus* ehk **inversioon** on ühe operandiga ehk *unaarne* loogikatehe.

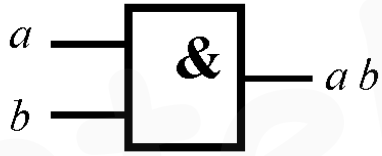
Loogikatehted **AND** ja **OR** on lihtsaimad "töötlusviisid" (16-st võimalikust), millega saab **kahest** digitaalsignaalist genereerida **ühe** uue digitaalsignaali.

Ülejäänuid vaadeldud (kahe sisendi/signaali) töötlusvõimalusi digitaaltöötlusel ei kasutata.

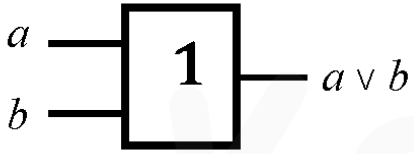
Need **3** ongi lihtsaimad / elementaarsed loogikatehted, mida omavahel kokkukombineerides saab koostada mistahes keerukusega digitaaltöötlust. Nendel kolmel elementaarsel *loogikatehtel* **NOT AND OR** põhineb kogu digitaalne signaalitöötlus kõikides digitaalseadmetes.

**JA**-elemendi *tingmärgid* loogikaskeemides:



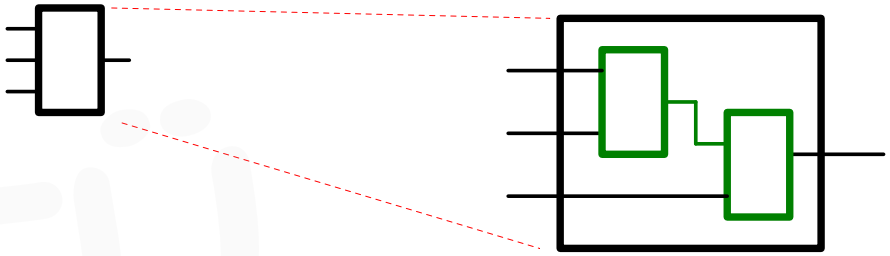

**VÕI**-elemendi *tingmärgid* loogikaskeemides:


Läheme veel sammuvõrra keerulisemaks:

vaatleme seadet millel on **3 sisendit** ja **1 väljund**.

Selline (ja ka rohkemate sisenditega) mistahes digitaalseade on alati koostatav piisava arvu **2 sisendiga** loogikaelementide ja **invertorite** sobiva kokkulülitamise teel:



**3 sisendiga** seade koostatud lülitusena **2 sisendiga** loogikaelementidest

Seega osutuvad digitaaltöötluse jaoks oluliseks ja piisavaks just **2 sisendiga** töötlevad elemendid.

## digitaaltehnika loogikatehete nimed ja nende tehtemärgid

| tehtemärk | tehte **nimi** ja selgitus |
|---|---|
| ¯ | loogiline **eitus** ehk **inversioon**<br>*nimetust "EI-tehe" tavaliselt ei kasutata* |
| ∧ | loogiline **korrutamine** ehk **konjunktsioon** ehk **JA**-tehe<br>( *aritmeetilise korrutamise* analoog loogikas ) |
| ∨ | loogiline **liitmine** ehk **disjunktsioon** ehk **VÕI**-tehe<br>( *aritmeetilise liitmise* analoog loogikas ) |

**JA**-tehte märgina kasutatakse ka sümbolit *'ampersand'*: **&** ( **&** ≡ ∧ )

**VÕI**-tehte märgina kasutatakse ka sümbolit **+** ( **+** ≡ ∨ )

**Inversioon**i võidakse tähistada samuti erinevalt: $\overline{\mathbf{A}}$ ≡ **'A** ≡ **A'**

*Aritmeetilise liitmise* tehtemärki **'+'** sobib kasutada **VÕI**-tehte ehk *disjunktsiooni* tehtemärgina sellepärast, et *disjunktsioon* on "tavalise" aritmeetilise liitmise analoog loogikas.

*loogikatehteid* on tegelikult rohkem (kokku 6 tk.) kuid digitaaltehnikas kasutatakse nendest ainult kolme elementaarset: **JA VÕI EI**

Nende 3 tehte abil on digitaalseadmes võimalik suvaline kogum **1**-sid ja **0**-lle töödelda suvaliseks muuks **1**-de ja **0**-de kogumiks.