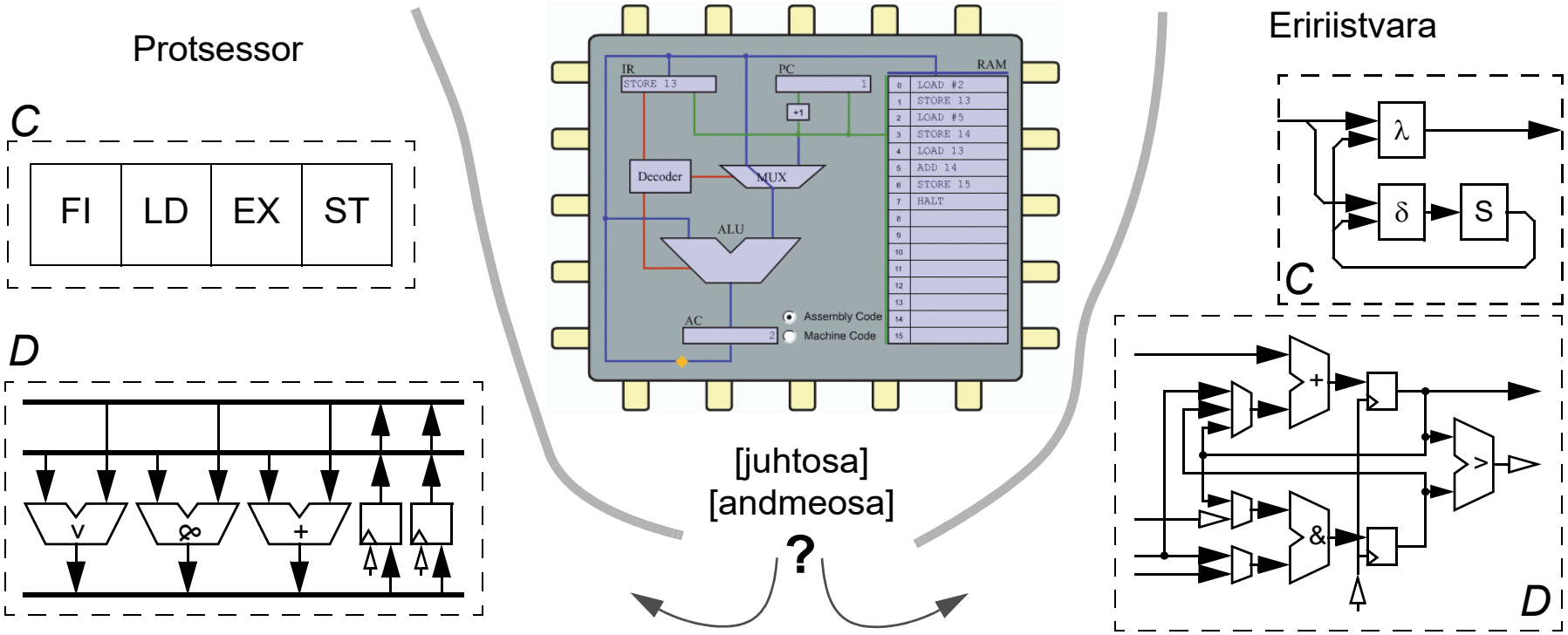
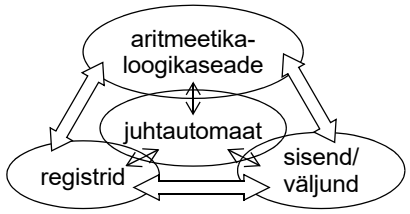
Aritmeetikatehete realiseerimine

Digitaalsüsteem – andme- ja juhtosa

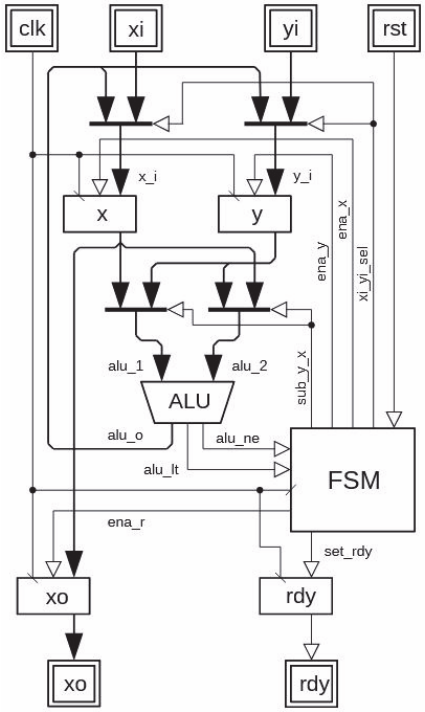


- **Andmeosa – operatsioonautomaat**
  - **töö andmetega – arvutused, salvestus, sisend/väljund**
- **Juhtosa – juhtautomaat**
  - **operatsioonide järjestamine vastavalt algoritmile**



Digitaalsüsteem – andme- ja juhtosa

Suurima ühisteguri leidmine (GCD)



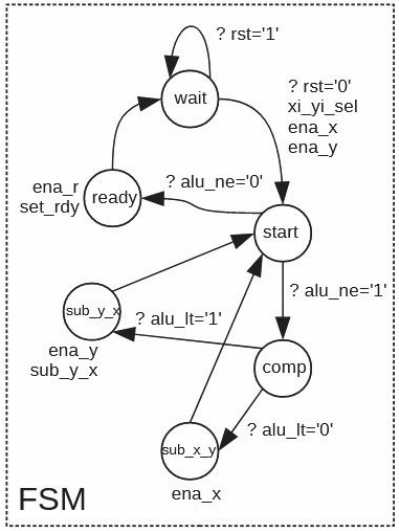
Code: gcd-rtl1.vhdl

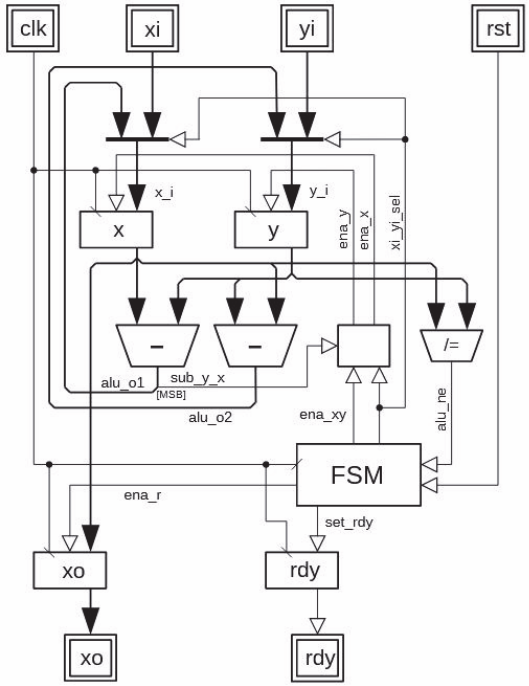
Single ALU ("–", "<", "/=")
[3 clock steps per iteration]

ASIC: 986 e.g. / 19.8 ns
FPGA: 50 SLC / 10.8 ns

Register-transfer level description with universal ALU (operation reuse).

Default value for the control signals is '0'





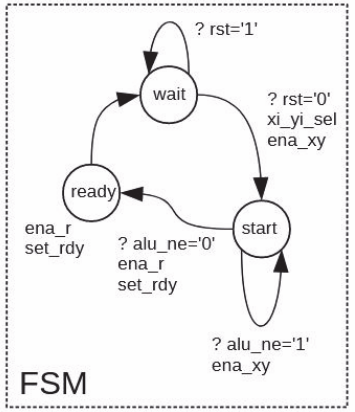
Code: gcd-rtl5.vhdl

2 subtracters, 1 comparator
[1 clock step per iteration]

ASIC: 915 e.g. / 20.0 ns
FPGA: 58 SLC / 8.0 ns

Register-transfer level description with out-of-order subtractions. Only data-path differs from RTL #3 & #4.

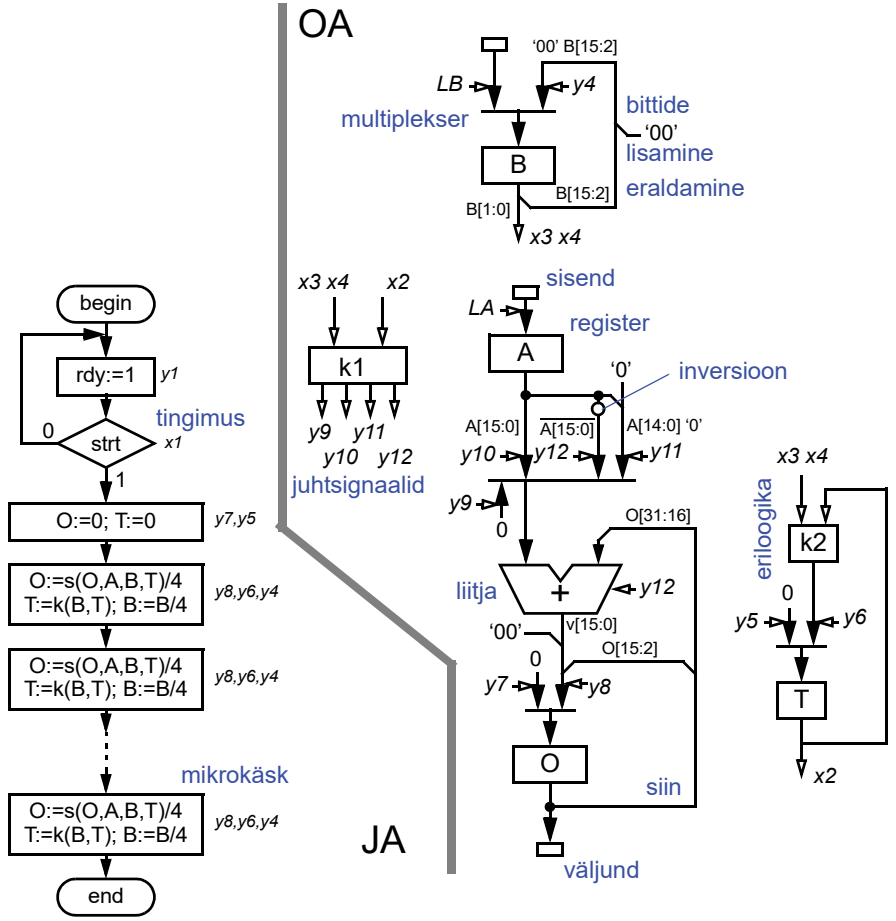
Default value for the control signals is '0'.



- **Sama algoritm, erinevad realisatsioonid [vt. ka https://ati.ttu.ee/~lrv/gcd/]**

Andmeosa

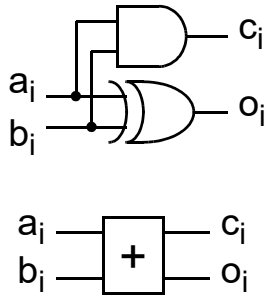
- **Töö andmetega – arvutused, edastus, salvestus, sisend/väljund**
  - **Arvutused**
    - **täitursõlmed – liitja, loendur, ALU jne**
    - **eriloogika, nihutamine**
  - **Edastus**
    - **siinid (andmesõnad)**
    - **multiplekserid**
  - **Salvestus**
    - **registrid / mäluplokid**
  - **Sisend/väljund**
    - **ühendus ülejäänud skeemiga**
- **Juhtosa**
  - **Operatsioonide tingimuslik järjestamine**
    - **mikroprogramm**
    - **mikrokäsud / tingimused**



Aritmeetikaoperatsioonid

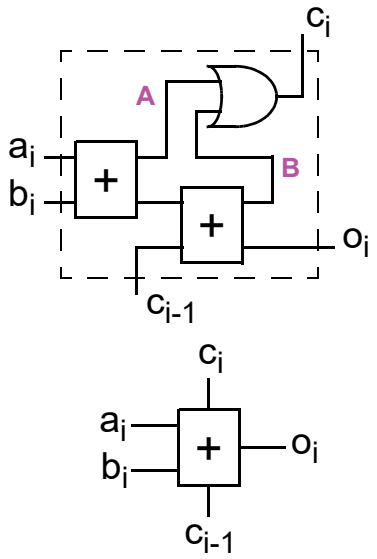
- **Kahendarvude liitmine**
  - **poolsummator – $(c_i,o_i) = a_i + b_i$**
    - **tulemus – $o_i = a_i \oplus b_i$**
    - **ülekanne – $c_i = a_i \,\&\, b_i$**

a_i	b_i	c_i	o_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



  - **täis-summator – $(c_i,o_i) = a_i + b_i + c_{i-1}$**
    - **tulemus – $o_i = a_i \oplus b_i \oplus c_i$**
    - **ülekanne – $c_i = (a_i \,\&\, b_i) \mid (a_i \,\&\, c_{i-1}) \mid (b_i \,\&\, c_{i-1})$**
      1 2 3
    - *38 (34) transistori*
    - **kaks pool-summaatorit ja VÕI-element**
    - *42 (38) transistori*

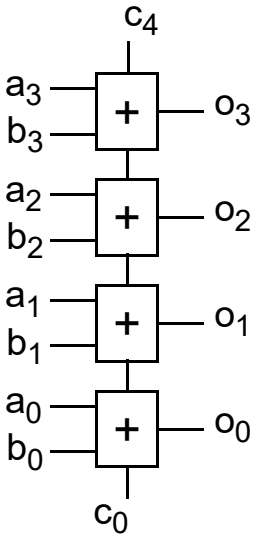
	a_i	b_i	c_{i-1}	c_i	o_i
	0	0	0	0	0
	0	0	1	0	1
	0	1	0	0	1
B 3	0	1	1	1	0
	1	0	0	0	1
B 2	1	0	1	1	0
A 1	1	1	0	1	0
A 1 2 3	1	1	1	1	1



Liitja ja lahutaja

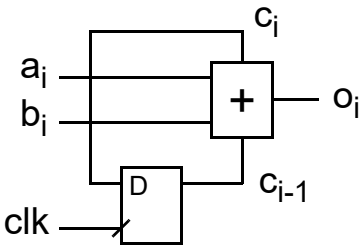
- **Summator e. liitja**
  - **Liitmine vs. lahutamine**

```
 119  01110111      119  01110111
+  9  00001001    -  9  00001001
[128] 01000000     [110] 01101110
```

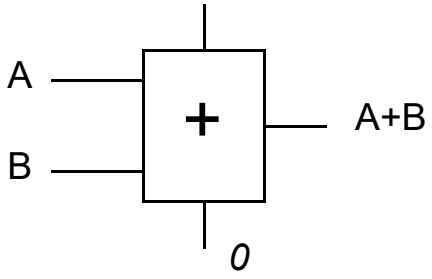


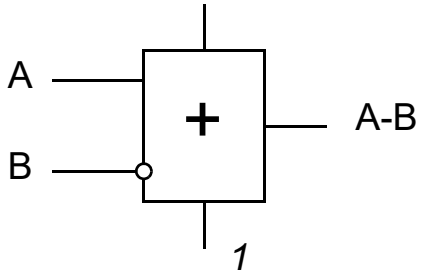
Paralleelne

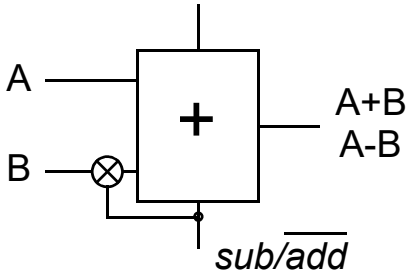
Järjestikuline



- **Lahutaja**
  - **täiendkood –** $O = A - B = A + \overline{B} + 1$
  - **liitja-lahutaja –** $O = n\ ?\ (A - B) : (A + B) = n\ ?\ (A + \overline{B} + 1) : (A + B + 0)$



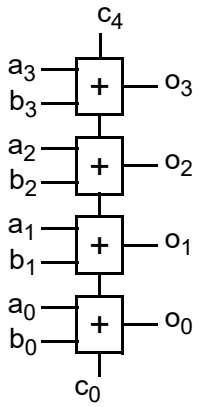




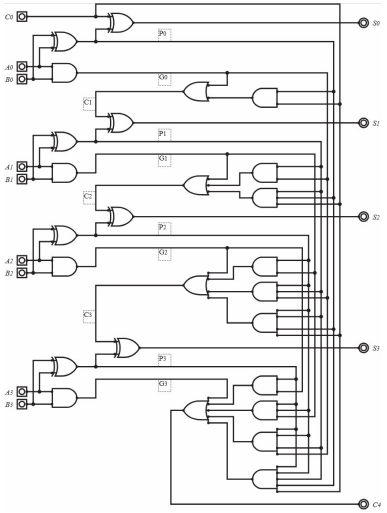
© Peeter Ellervee

Ülekande kiirendamine?

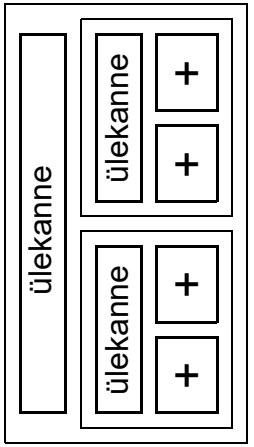
- Järjestikuline ülekanne
  - *ripple-carry adder*
  - Koren'i simulaator – 8 bitti & 16,1 ns



- Kiirendamine?
  - jagatud gruppideks
  - grupi sees ülekanne eraldi funktsioonina
- *carry-lookahead adder*
  - "generate" – $g_i = a_i \,\&\, b_i$; $g_{i+1} = a_{i+1} \,\&\, b_{i+1}$
  - "propagate" – $p_i = a_i \mid b_i$; $p_{i+1} = a_{i+1} \mid b_{i+1}$
  - "carry" – $c_i = g_i \mid (p_i \,\&\, c_{i-1})$; $c_{i+1} = g_{i+1} \mid (p_{i+1} \,\&\, c_i)$
  - –> $c_{i+1} = g_{i+1} \mid (p_{i+1} \,\&\, (g_i \mid (p_i \,\&\, c_{i-1}))) \ldots$
  - Koren'i simulaator – 8 bitti & 10,2 ns
  - Võimalik ka gruppide hierarhia

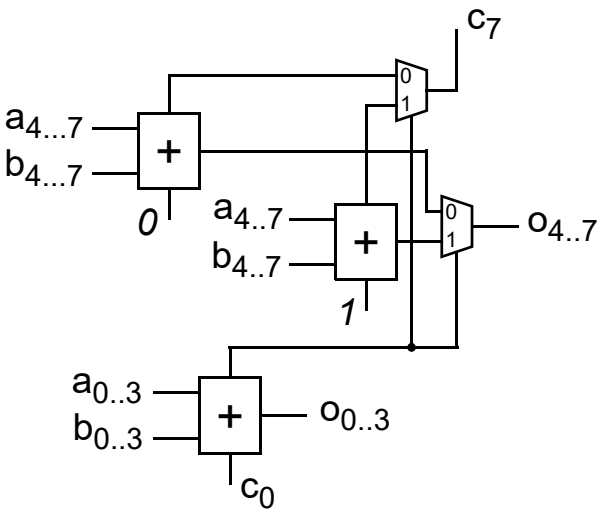


wikipedia.org

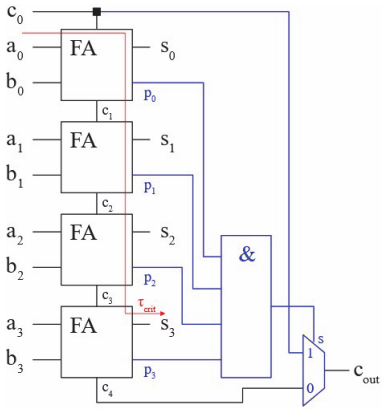


Ülekande kiirendamine?

- *carry-select adder*
  - spekulatiivne arvutus – vanemates järkudes leitakse summa ja ülekanne nii 0 kui 1 jaoks, tulemus valitakse multipleksoriga, kui ülekanne alt “lõpuks kohale jõuab”
  - Koren’i simulaator – 8 bitti & 11,3 ns
  - Võimalik ka gruppide hierarhia



- *carry-skip adder*
  - kui ülekanne plokis levib (p_i-d & AND), siis ülekanne edasi on sama, mis ülekanne plokki (c_0)
  - Koren’i simulaator – 8 bitti & 11,3 ns
    - eeldab rohkem plokke, et kiirendamine oleks märgatav
  - Võimalik ka gruppide hierarhia



wikipedia.org

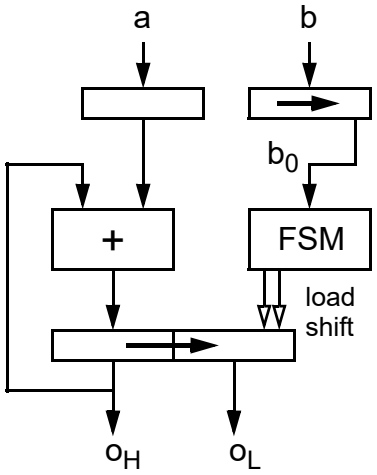
Korrutamine

- **Järjestikuline – järk-järgu kaupa tsükliliselt**
  - **akumulaator ja juhtautomaat**
  - **mitu järku korraga – juhtloogika lisaks**
- **Liitmine ja nihutamine**
  - **noorimatest järkudest alates**

```
   00011000 * 00001101
   -------------------
1.           00000000
2.          00000000
3.         00000000
4.        00001101
5.       00001101
6.      00000000
7.     00000000
8.    00000000
   -------------------
      000000100111000
```

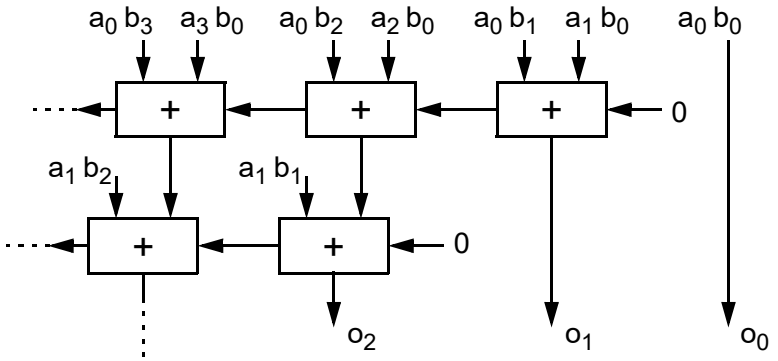
vanimatest järkudest alates

```
00011000 * 00001101
-------------------
    00000000
     00000000
      00000000
       00001101
        00001101
         00000000
          00000000
           00000000
-------------------
     00000100111000
```



Korrutamine

- **Paralleelne – kombinatsioonskeem**
  - **m*n bitti –**
    m n-bitist summaatorit ja m*n JA-elementi
  - **kombinatsioonskeem võib olla liiga suure viitega –**
    abiregistrid & konveier (pipeline)
- **Polünoomide korrutamine?!**
  - $A = a_{n-1}*2^{n-1}+a_{n-2}*2^{n-2}+...+a_2*2^2+a_1*2^1+a_0*2^0$
  - $B = b_{n-1}*2^{n-1}+b_{n-2}*2^{n-2}+...+b_2*2^2+b_1*2^1+b_0*2^0$
  - $A * B = a_{n-1}*2^{n-1}*(b_{n-1}*2^{n-1}+b_{n-2}*2^{n-2}+...+b_2*2^2+b_1*2^1+b_0*2^0)+$
    $a_{n-2}*2^{n-2}*(b_{n-1}*2^{n-1}+b_{n-2}*2^{n-2}+...+b_2*2^2+b_1*2^1+b_0*2^0)+...$
    $a_2*2^2*(b_{n-1}*2^{n-1}+b_{n-2}*2^{n-2}+...+b_2*2^2+b_1*2^1+b_0*2^0)+$
    $a_1*2^1*(b_{n-1}*2^{n-1}+b_{n-2}*2^{n-2}+...+b_2*2^2+b_1*2^1+b_0*2^0)+$
    $a_0*2^0*(b_{n-1}*2^{n-1}+b_{n-2}*2^{n-2}+...+b_2*2^2+b_1*2^1+b_0*2^0) =$
    $a_{n-1}*b_{n-1}*2^{2*n-2}+a_{n-1}*b_{n-2}*2^{2*n-3}+...+a_0*b_2*2^2+a_0*b_1*2^1+a_0*b_0*2^0$

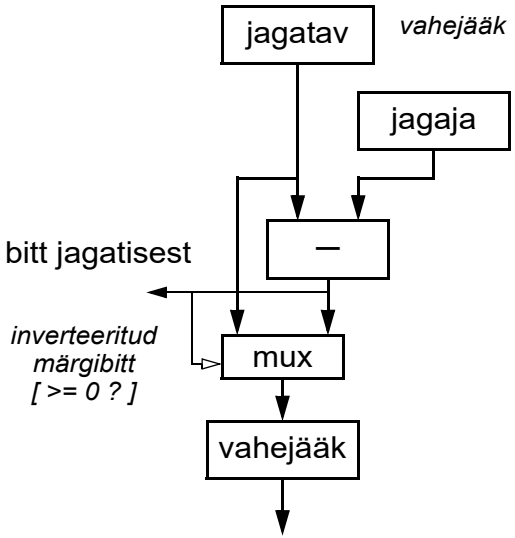


- **Negatiivsete arvude vahetul korrutamisel peab arvestama täiendkoodidega**

Jagamine

- **Tavaliselt järjestikuliselt (tsükliliselt)**
  - **akumulaator, juhtloogika ja juhtautomaat**

```
23 / 5 = 4 jääk 3

00010111 / 00000101 = 100 jääk 11
 -0101
  ------
      11
```



- **Negatiivsete arvude vahetul jagamisel peab arvestama täiendkoodidega**

- **Korrutamisel ja jagamisel võib vaja minna ümardamist**
  - **Vt. Arvusüsteemid (3) – Püsi- ja ujukoma-arvud**

© Peeter Ellervee

Ületäitumine?

- Täiendkood
  - 8 bitti: -128...+127
  - 125+5=? [130]

```
 125  01111101
+  5  00000101
[130] 10000010 == -126
```

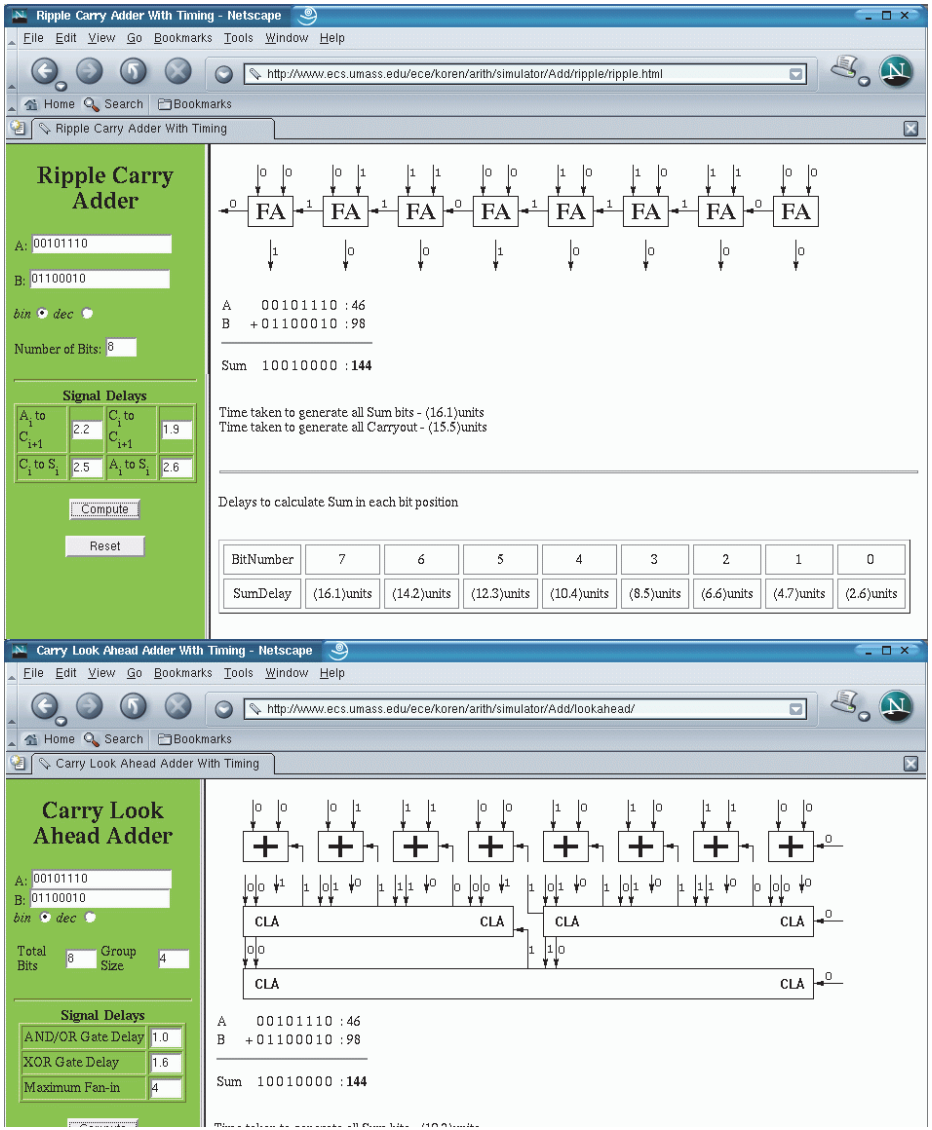
- Arvude skaala

```
  -1  11111111
      ........
-128  10000000
+127  01111111
      ........
   0  00000000
```

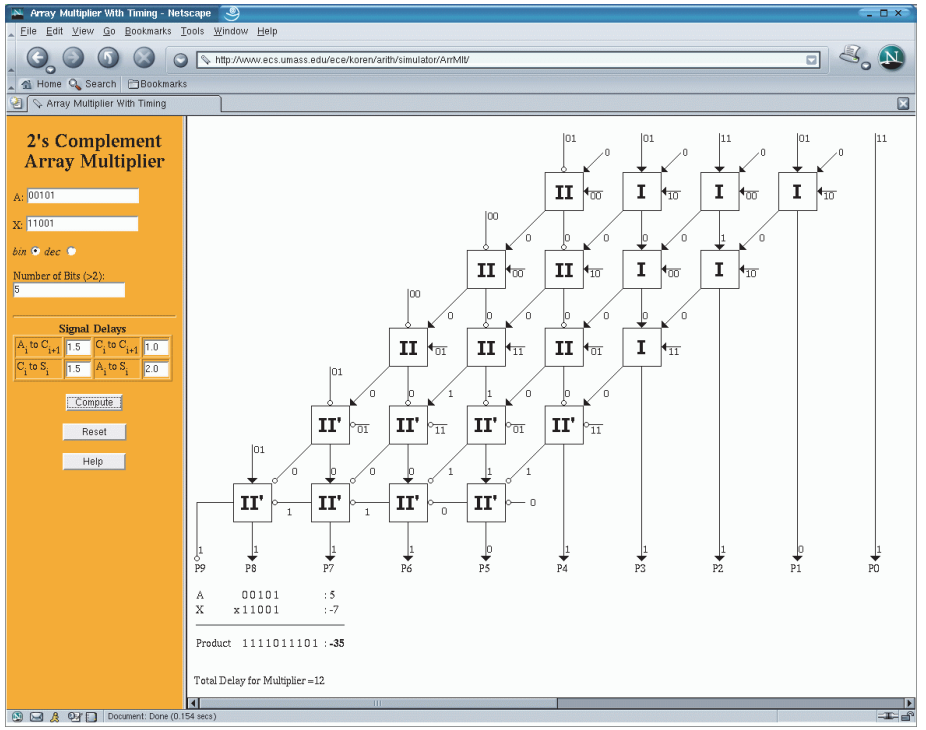
- Ekstra märgibitt
  - 00 või 11 – OK

```
   75  001001011
+  15  000001111
  [90] 001011010

  125  001111101
+   5  000000101
 [130] 010000010

  -75  110110101
+ -15  111110001
 [-90] 110100110

 -125  110000011
+  -5  111111011
[-130] 101111110
```

© Peeter Ellervee



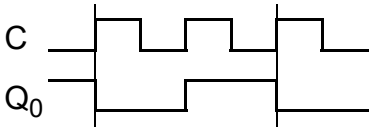
http://www.ecs.umass.edu/ece/koren/arith/simulator/



Loendur kui juhtautomaat [#1]

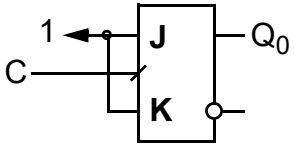
- **Loendur – primitiivne generaator – S≠∅, I=∅, O=∅ (O≡S), δ:S→S, λ≡∅**

2-nd loendur
Jada – 0, 1

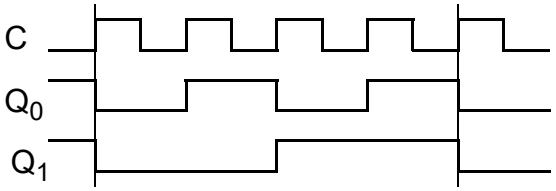


s_t	s_{t+1}	JK_{t+1}
0	1	1-
1	0	-1

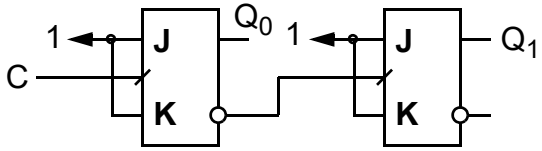
J=K=1



4-nd loendur
Jada – 0, 1, 2, 3



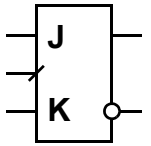
Kaks loendurit järjest?



Takti hilistumine!!!

Automaat?
Olekud – 00, 01, 10, 11 [Q1 Q0]

JK-triger

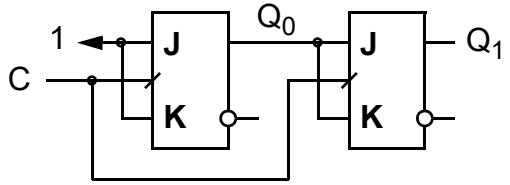


Q_t	Q_{t+1}	J	K
0	0	0	-
0	1	1	-
1	0	-	1
1	1	-	0

s_t	s_{t+1}	JK_{t+1}
00	01	0- 1-
01	10	1- -1
10	11	-0 1-
11	00	-1 -1

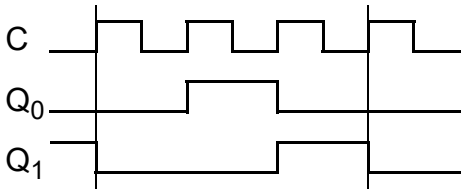
$J_0=K_0=1$

$J_1=K_1=Q_0$

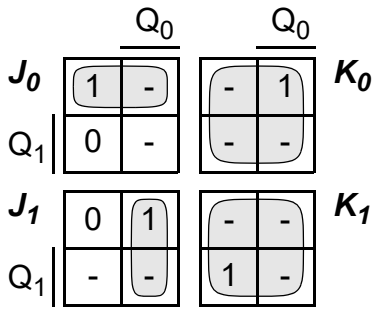


Loendur kui juhtautomaat [#2]

3-nd loendur
Jada – 0, 1, 2
Olekud – 00, 01, 10 [Q1 Q0]



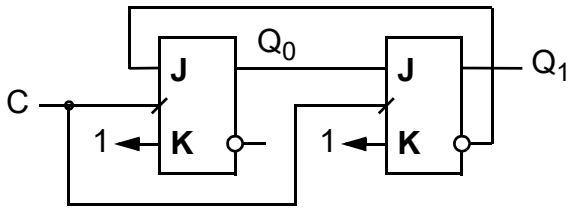
s_t	s_{t+1}	JK_{t+1}
00	01	0- 1-
01	10	1- -1
10	00	-1 0-



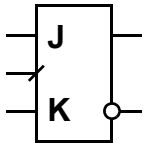
$K_0=K_1=1$

$J_0=\overline{Q}_1$

$J_1=Q_0$



JK-triger



Q_t	Q_{t+1}	J	K
0	0	0	-
0	1	1	-
1	0	-	1
1	1	-	0

5-nd loendur
Jada – 0, 1, 2, 3, 4
Olekud – 000, 001, 100, 011, 100 [Q2 Q1 Q0]

s_t	s_{t+1}	JK_{t+1}
000	001	0- 0- 1-
001	010	0- 1- -1
010	011	0- -0 1-
011	100	1- -1 -1
100	000	-1 0- 0-

$(J_0=\overline{Q}_2 \quad K_0=1)$

$J_0=K_0=\overline{Q}_2$

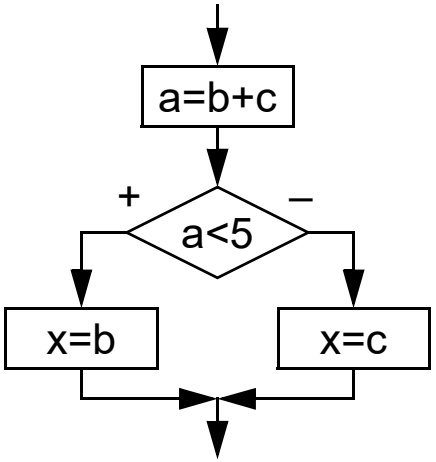
$J_1=K_1=Q_0$

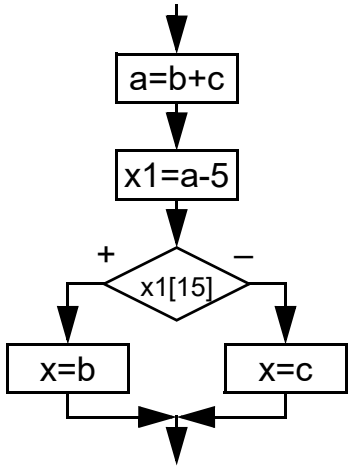
$J_2=Q_1Q_0 \quad K_2=1$

Plokkskeemi genereerimine algoritmist

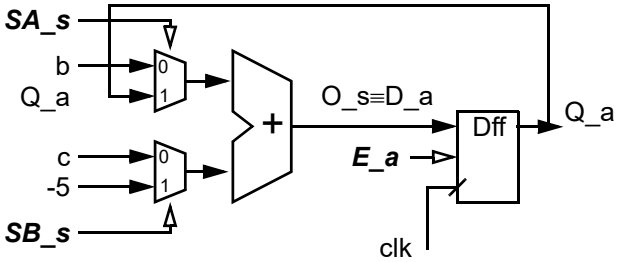
```
...
a = b + c ;
if ( a < 5 )    x = b ;
else            x = c ;
...
```

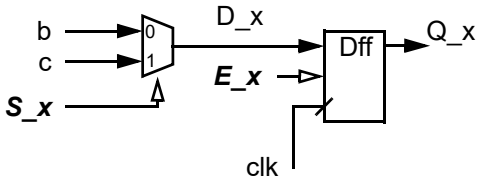
- **Andme-osa**
  - **Muutujad e. registrid**
    - **a (x1), b, c, x**
  - **Operatsioonid e. ALU-d – '+', '<'**
    - **kombinatsioonskeemid**
    - **antud juhul üks liitja**
    - **a < 5 ≡ a+(-5) < 0 [märgibitt!]**
  - **Omistamised e. multiplekserid**
- **Juht-osa**
  - **Algoritm**
    - **juhtsignaalid (juht)automaadist**
    - **kontrollsignaalid (juht)automaati**





Andmeosa





a=b+c
E_a=1; E_x=0; S_x=*;
SA_s=0; SB_s=0;

x1=a-5
E_a=1; E_x=0; S_x=*;
SA_s=1; SB_s=1;

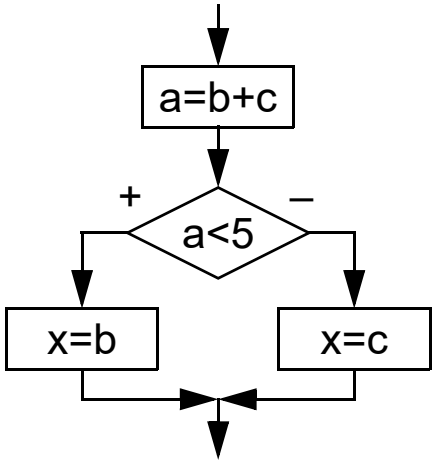
x=b
E_a=0; E_x=1; S_x=0;
SA_s=*; SB_s=*;

x=c
E_a=0; E_x=1; S_x=1;
SA_s=*; SB_s=*;

© Peeter Ellervee

Plokkskeem ja juhtautomaadid

```
...
a = b + c ;
if ( a < 5 )    x = b ;
else            x = c ;
...
```

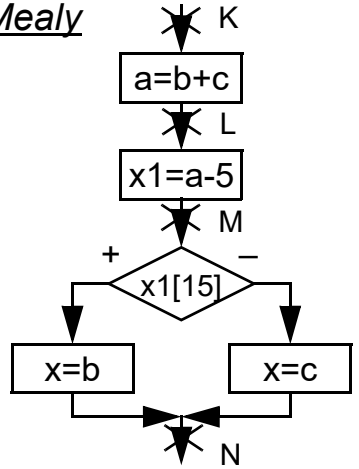


Mealy: i^t – x1[15] (Q_a[15])

Moore: i^t – O_s[15]

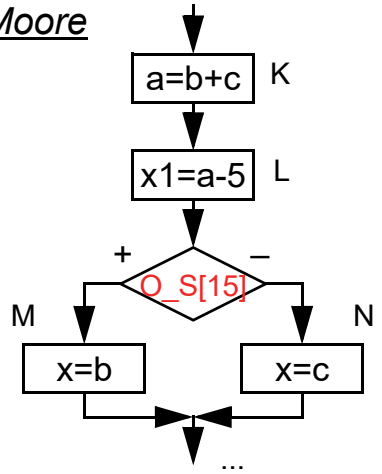
o^t – E_a, E_x, S_x, SA_s, SB_s

Mealy



i^t	s^t	s^{t+1}	o^t
...	...	K	.....
-	K	L	10-00
-	L	M	10-11
0	M	N	011--
1		N	010--
...	N	...	...

Moore



i^t	s^t	s^{t+1}	o^t
...	...	K	.....
-	K	L	10-00
0	L	N	-0-11
1		M	
-	M	...	010--
-	N	...	011--

© Peeter Ellervee