



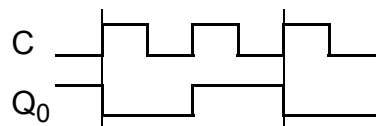
Juht-automaadid digitaalsüsteemides

- **Automaadi olekudiagrammi / tabeli genereerimine (süntees)**
 - tabeli süntees plokk-skeemist (plokk-diagrammist)
 - plokk-skeemi genereerimine kõrgtaseme keeltest
 - plokk-skeemi genereerimine algoritmist
 - andme-osa / operatsioon-automaadi süntees – struktuuri genereerimine
 - juht-osa / juht-automaadi süntees – operatsioonid järjestused
- ***Automaadi süntees olekudiagrammist / tabelist***
 - *sisendite / väljundite kodeerimine*
 - *olekute kodeerimine*
 - *siirde- ja väljundfunktsiooni süntees ja minimeerimine*
- **Seos ülejäänud süsteemiga, st. andmeosaga, mäludega jne.?**

Loendur kui automaat [#1]

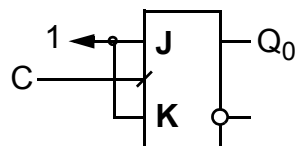
- Loendur – primitiivne generaator – $S \neq \emptyset$, $I = \emptyset$, $O = \emptyset$ ($O \equiv S$), $\delta: S \rightarrow S$, $\lambda \equiv \emptyset$

2-nd loendur Jada – 0, 1

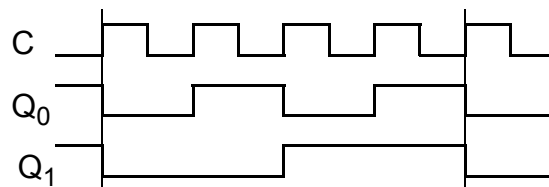


s_t	s_{t+1}	JK_{t+1}
0	1	1-
1	0	-1

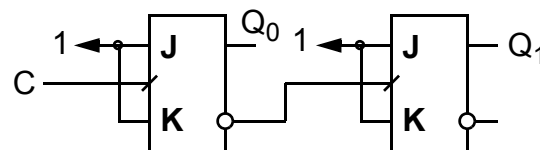
$J=K=1$



4-nd loendur Jada – 0, 1, 2, 3



Kaks loendurit järjest?

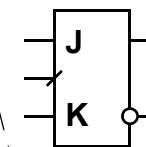


Takti hilistumine!!!

Automaat?

Olekud – 00, 01, 10, 11 [Q1 Q0]

JK-triger

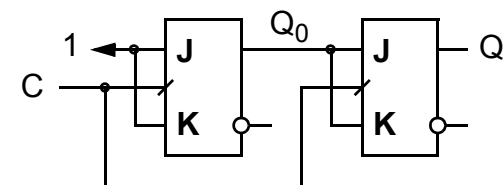


Q_t	Q_{t+1}	J	K
0	0	0	-
0	1	1	-
1	0	-	1
1	1	-	0

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

$J_0=K_0=1$

$J_1=K_1=Q_0$

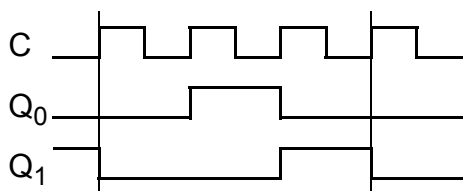


Loendur kui automaat [#2]

3-nd loendur

Jada – 0, 1, 2

Olekud – 00, 01, 10 [Q1 Q0]

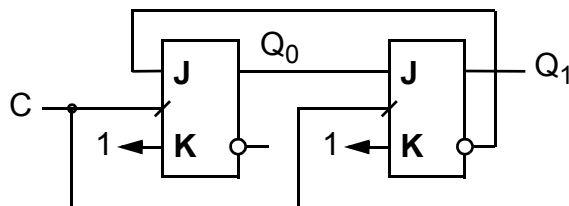


s_t	s_{t+1}	JK_{t+1}		
00	01	0- 1-	J_0	K_0
01	10	1- -1	J_1	K_1
10	00	-1 0-		

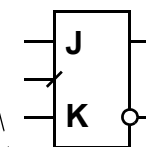
$$K_0 = K_1 = 1$$

$$J_0 = \bar{Q}_1$$

$$J_1 = Q_0$$



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-

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

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

$$J_1 = K_1 = Q_0$$

$$J_2 = Q_1 Q_0 \quad K_2 = 1$$

Loendur kui automaat [#3]

- Pseudojuhuarvu generaator (LFSR) – 7, 5, 1, 2, 4, 3, 6, 7, 5, ...

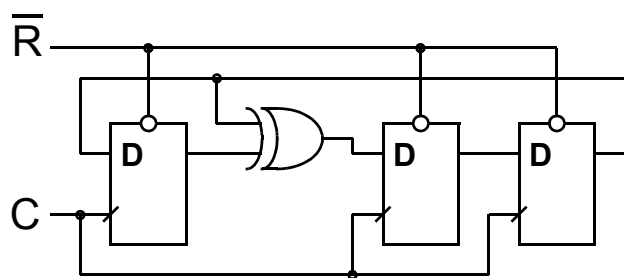
s_t	s_{t+1}	D_{t+1}
111	101	101
101	001	001
001	010	010
010	100	100
100	011	011
011	110	110
110	111	111

$[Q_3 \ Q_2 \ Q_1]$

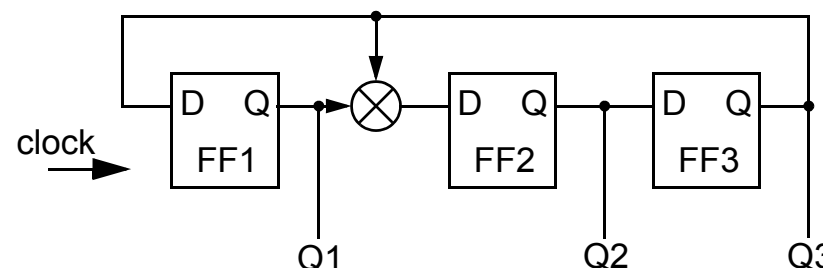
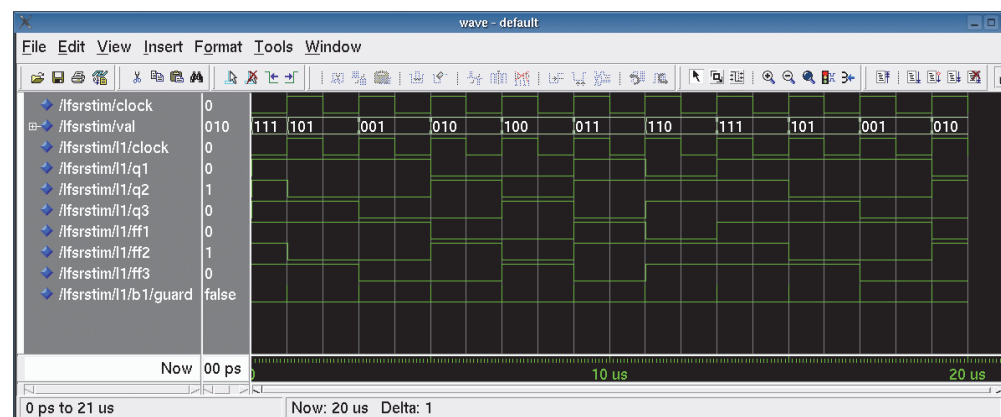
$$D_1 = Q_3$$

$$D_2 = Q_3 \otimes Q_1$$

$$D_3 = Q_2$$



Vrdl. LFSR näidet VHDL-s



VHDL & kodutöö #2

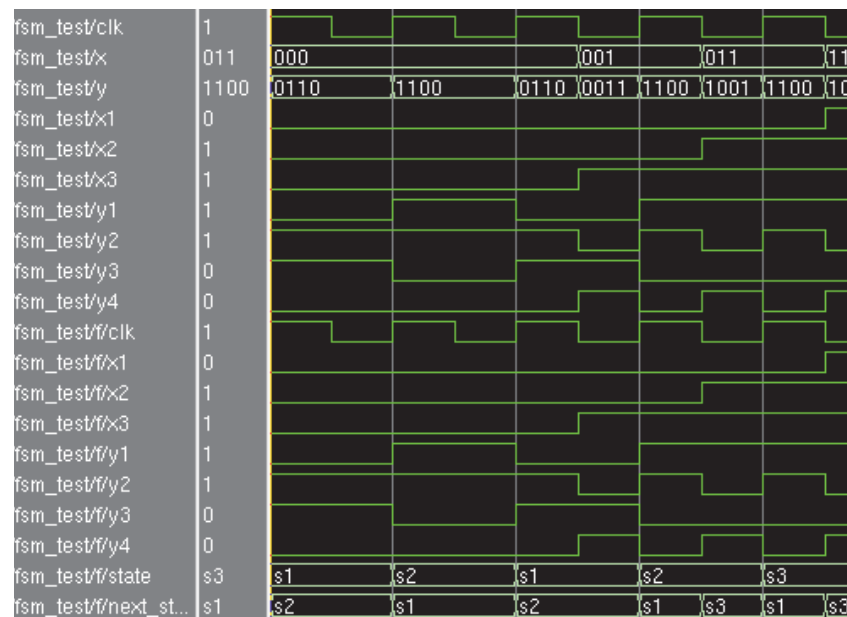
• Testpink – kõik siirded tuleb läbida

```
process begin
    wait on clk until clk='0';    x3<='0';    -- S1 -> S2
    wait on clk until clk='0';    x2<='0';    -- S2 -> S1
    wait on clk until clk='0';    x3<='1';    -- S1 -> S2
    wait on clk until clk='0';    x2<='1';    -- S2 -> S3
    wait on clk until clk='0';    x1<='1';    -- S3 -> S3
    wait on clk until clk='0';    x1<='0';    -- S3 -> S1
end process;
```

• Automaat – käitumuslik mudel

```
-- Next state and output functions
process (x1, x2, x3, state) begin
    case state is
        when s1 => if x3='0' then
            next_state<=s2; y1<='0'; y2<='1'; y3<='1'; y4<='0';
        else
            next_state<=s2; y1<='0'; y2<='0'; y3<='1'; y4<='1';
        end if;
        ...
    end case;
end process;

-- State register
process begin
    wait on clk until clk='1';    state<=next_state;
end process;
```



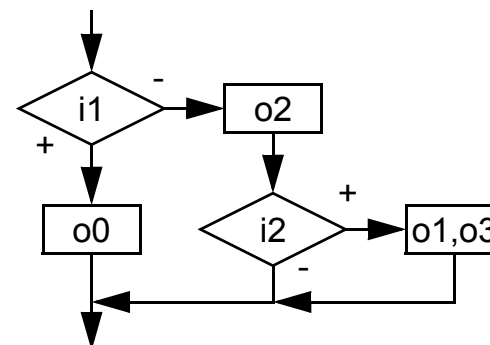
Plokskeemi genereerimine

- Automaatne genereerimine kõrgtaseme kirjeldusest
 - programmeerimiskeel või riistvara kirjelduskeel

```

...
if (inp=="-0--") {
    outp= "0010";
    if (inp=="--1-") outp= "0101";
}
else outp= "1000";
...

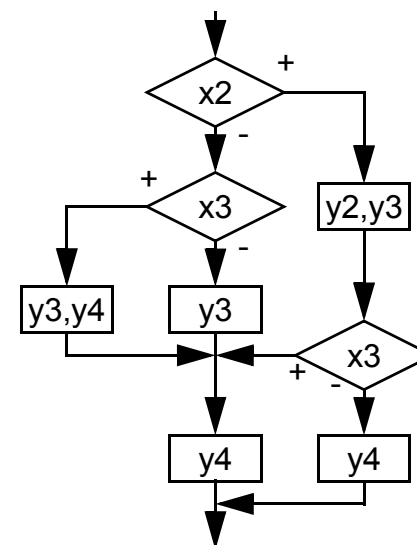
```



```

...
if (x2) {
    out (y2,y3);
    if (!x3) {
        out (y4); goto L1;
    }
}
else {
    if (x3) out (y3,y4);
    else out (y3);
}
out (y4);
...

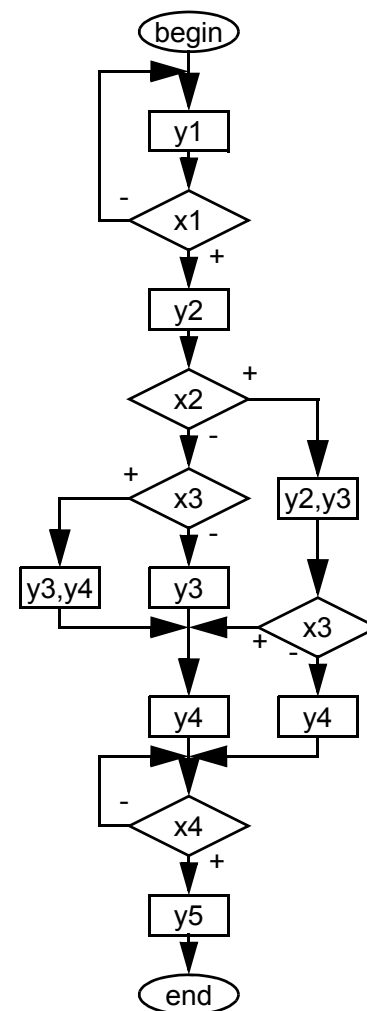
```



Plokkskeemi genereerimine – näide

```

process fsm (x1,x2,x3,x4,y1,y2,y3,y4,y5)
  bit in x1,x2,x3,x4;  bit out y1,y2,y3,y4,y5;
{
  while (!x1)  out (y1);
  out (y2);
  if (x2) {
    out (y2,y3);
    if (!x3) { out (y4); goto L1; }
  }
  else { if (x3) out (y3,y4);  else  out (y3); }
  L1: while (!x4);
  out (y5);
}
  
```



Plokkskeemi genereerimine algoritmist

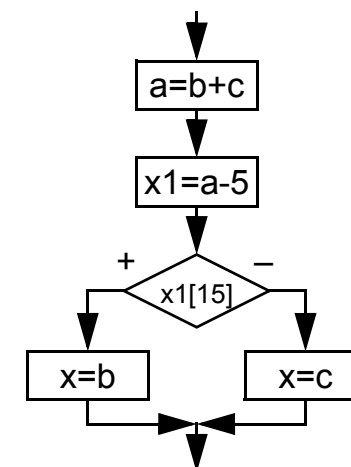
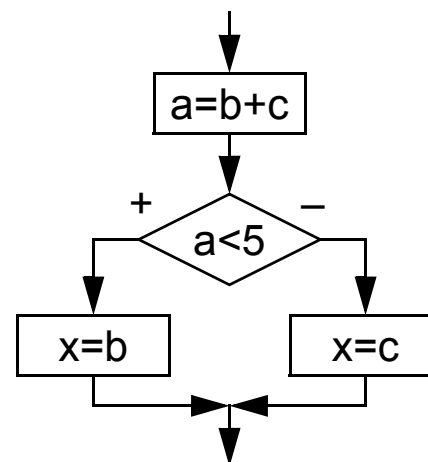
```
...
a = b + c ;
if ( a < 5 )      x = b ;
else              x = c ;
...
```

- **Andme-osa**

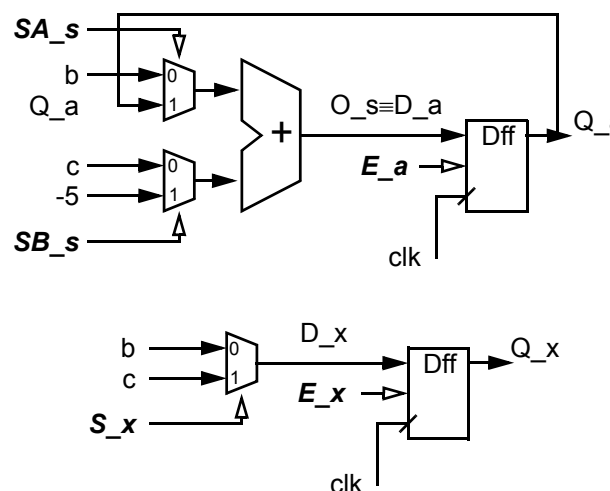
- **Muutujad e. registrid**
 - a (x1), b, c, x
- **Operatsioonid e. ALU-d – '+', '<'**
 - kombinatsioonskeemid
 - antud juhul üks liitja
 - $a < 5 \equiv a + (-5) < 0$ [märgibitt!]
- **Omistamised e. multiplekserid**

- **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;
```

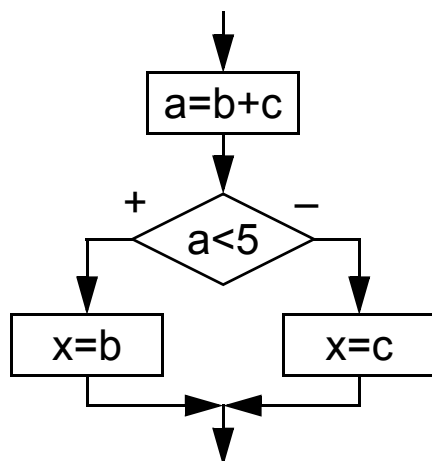
```
x1=a-5
E_a=1; E_x=0; S_x=*;
SA_s=1; SB_s=1;
```

```
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=*
```

Plokkskeem ja juhtautomaadid

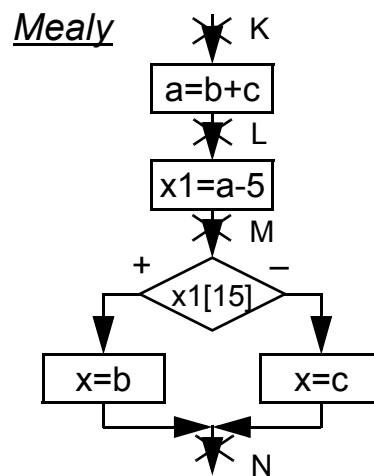
```
...
a = b + c ;
if ( a < 5 )   x = b ;
else          x = c ;
...
```



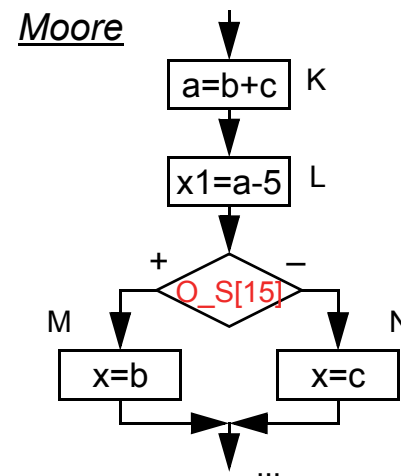
Mealy: $i^t - x1[15]$ (Q_a[15])

Moore: $i^t - O_s[15]$

$o^t - E_a, E_x, S_x, SA_s, SB_s$



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	...	...

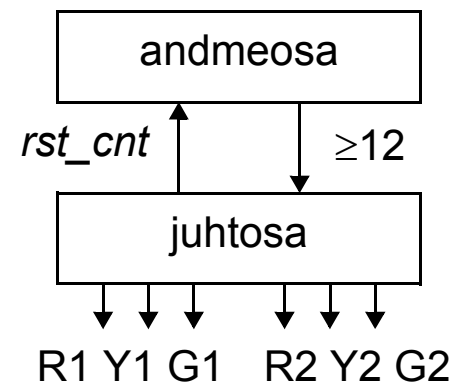


i^t	s^t	s^{t+1}	o^t
...	...	K	
-	K	L	10-00
0	L	N	-0-11
1		M	
-	M	...	010--
-	N	...	011--

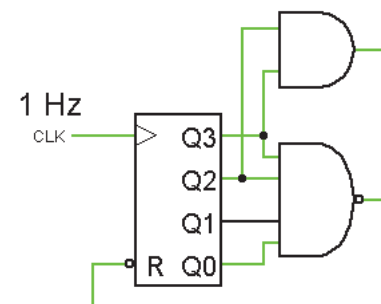
Valgusfoor kui digitaalsüsteem – näide #1

- Digitaalsüsteem – andmeosa + juhtosa
- Kogutsükkel 30 sek., andurid puuduvad
 - roheline 12 sek. punane
 - kollane 3 sek. kollane+punane
 - punane 12 sek. roheline
 - kollane+punane 3 sek. kollane
- Juhtosa – automaat
 - $I = \{<12, \geq 12\}$, $O = \{R1, Y1, G1, R2, Y2, G2, rst_cnt(?)\}$
- Andmeosa – loendur (0...14)
 - $0...14 \rightarrow 4$ bitti (0...15!)
 - asünkroonne nullimine kui loendur == 15 (e. 4-NAND)
 - 12 sek. == 0...11 / 3 sek. == 12...14 e. ≥ 12
 - $\geq 12 == 1100 + 1101 + 1110$ (+1111 määramatusena)
 - $\geq 12 == 11--$ (e. 2-AND)
 - rst_cnt vajalikkus? – sõltub juhtosa “tarkusest”

<http://mini.pld.ttu.ee/~lrv/IAY0150/tlc-datapath.txt>



Andmeosa



Valgusfoor – juhtosa

- roheline 12 sek. punane   cnt=0...11
- kollane 3 sek. kollane+punane   cnt=12...14
- punane 12 sek. roheline   cnt=0...11
- kollane+punane 3 sek. kollane   cnt=12...14

```

while ( !≥12 )           // S1
    set ( G1, R2 );
while ( ≥12 )             // S2
    set ( Y1, Y2, R2 );
while ( !≥12 )           // S3
    set ( R1, G2 );
while ( ≥12 )             // S4
    set ( Y1, R1, Y2 );

```

i^t	s^t	s^{t+1}	o^t
$!≥12$	S1	S1	G1 , R2
$≥12$		S2	
$!≥12$	S2	S3	Y1 , Y2 , R2
$≥12$		S2	
$!≥12$	S3	S3	R1 , G2
$≥12$		S4	
$!≥12$	S4	S1	Y1 , R1 , Y2
$≥12$		S4	

Valgusfoor – juhtosa

i^t		s^t		s^{t+1}		J K	D	o^t	
$! \geq 12$	0	S1	00	S1	00	0- 0-	00	G1, R2	001 100
≥ 12	1			S2	01	0- 1-	01		
$! \geq 12$	0	S2	01	S3	11	1- -0	11	Y1, Y2, R2	010 110
≥ 12	1			S2	01	0- -0	01		
$! \geq 12$	0	S3	11	S3	11	-0 -0	11	R1, G2	100 001
≥ 12	1			S4	10	-0 -1	10		
$! \geq 12$	0	S4	10	S1	00	-1 0-	00	Y1, R1, Y2	110 010
≥ 12	1			S4	10	-0 0-	10		

- Väljundid**

$$Y1 = Y2 = q1 \text{ xor } q2$$

$$R1 = q1$$

$$R2 = q1'$$

$$G1 = q1' \ q2'$$

$$G2 = q1 \ q2$$

- JK-trigerid**

$$j1 = i' \ q2$$

$$k1 = i' \ q2'$$

$$j2 = i \ q1'$$

$$k2 = i \ q1$$

- D-trigerid**

$$d1 = i \ q1 + i' \ q2$$

$$d2 = i \ q1' + i' \ q2$$

$$\# \text{ Optimeerimist...}$$

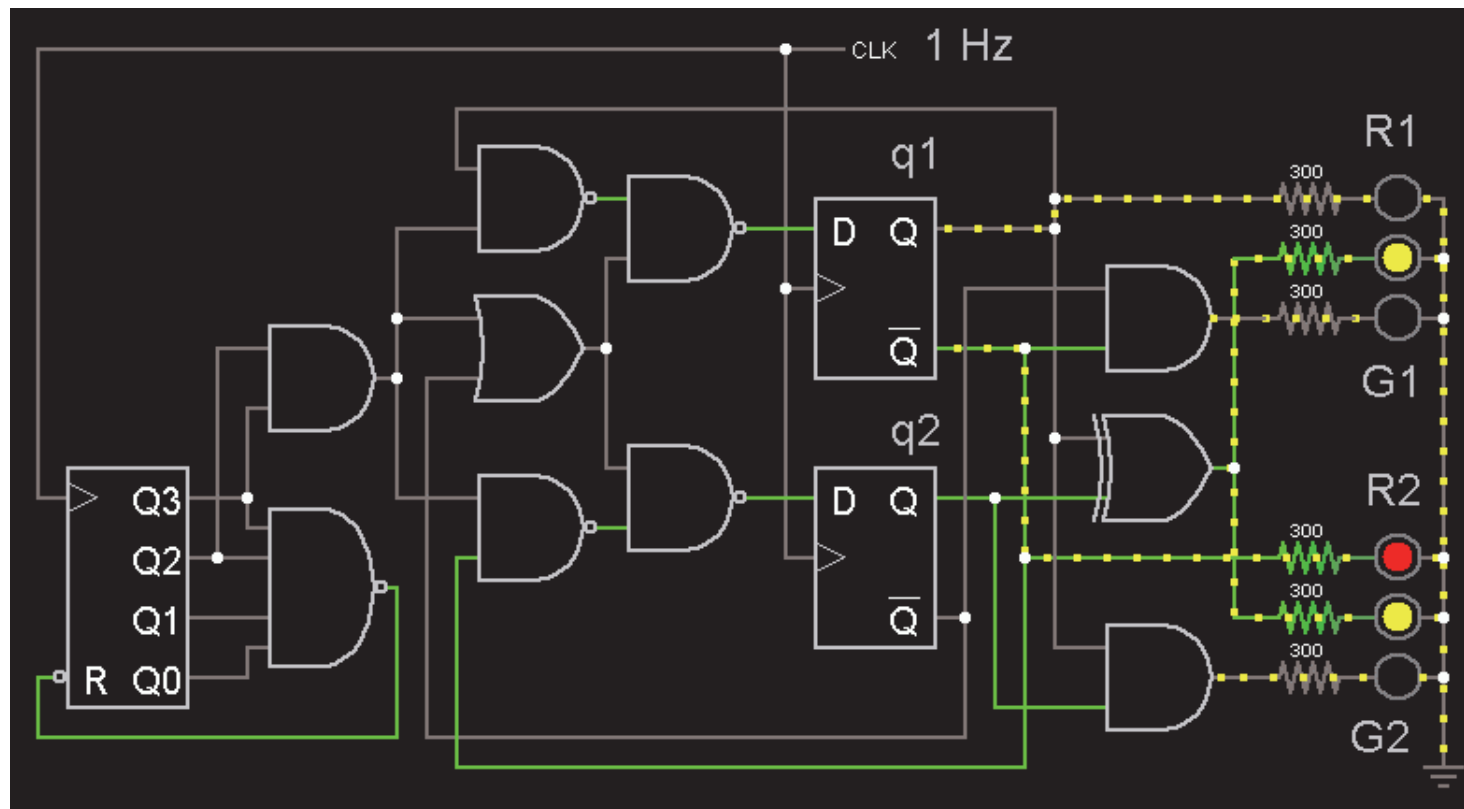
$$i' \ q2 = (i + q2')'$$

$$k = i + q2'$$

$$d1 = ((i \ q1)' \ k)'$$

$$d2 = ((i \ q1')' \ k)'$$

Valgusfoor – tulemus



<http://mini.pld.ttu.ee/~lrv/IAY0150/tlc-applet.txt>



Näide #2 – märgita täisarvude korrutamine

- Lihtne algoritm

- andmeosa

- liitmine & nihutamine/nihutamised
 - 3 registrit – 2 tegurit, korrutis

- juhtosa

- iteratsioonid
 - teguri biti kontroll
 - summaatori ja registrite juhtimine

```

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

- Kaks bitti korraga?

- vähem iteratsioone – kiirem

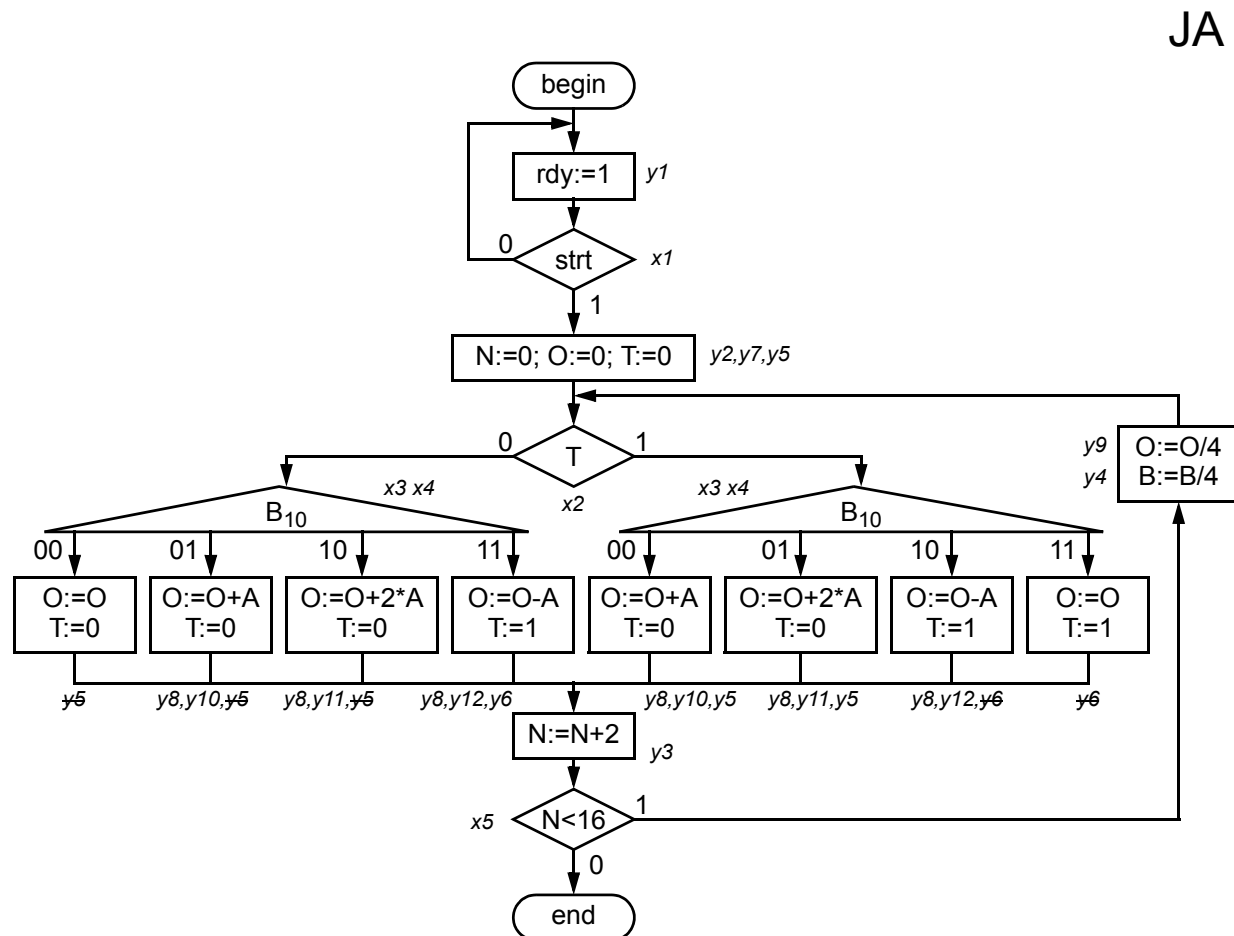
- keerukam algoritm

- keerukam andmeosa
 - keerukam juhtosa

```

00011000 * 00001101
-----
1.          00000000    00011000 == * 0
2.          00011010..   00011000 == * 2
3.          00001101.... 00011000 == * 1
4.          00000000..... 00011000 == * 0
-----
00000100111000
  
```

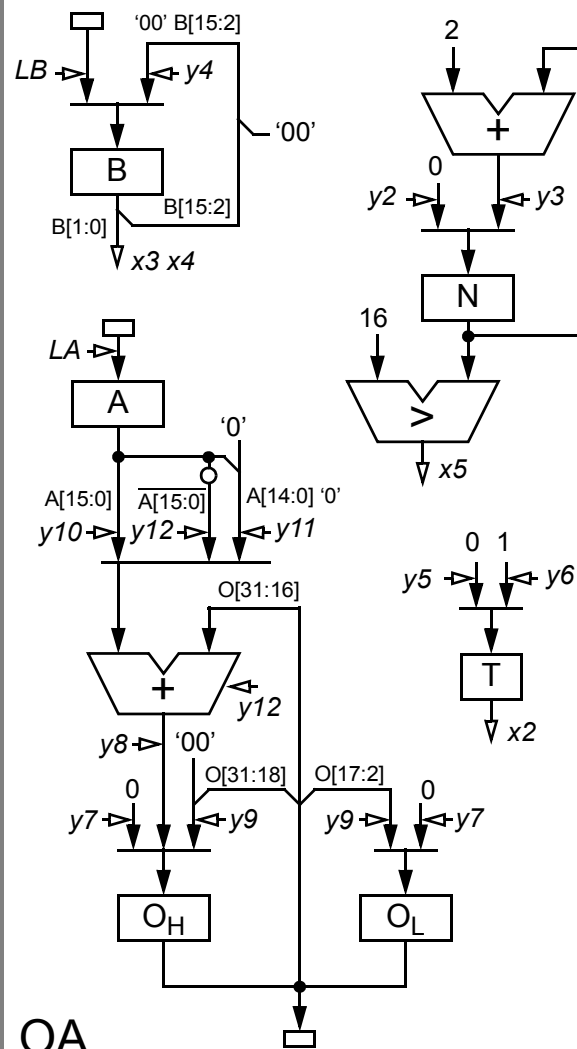

Märgita täisarvude korrutamine – realisatsiooni näide



JA Moore: i 5, o 12, s 12, t 28; Mealy: i 5, o 12, s 4, t 13 (tagasiside OK)

OA 2 sum., 1 võrdl., 5 (6) mux, 5 (6) reg.

3 takti iteratsiooni kohta, kokku 24 (+1) takti





Näide #3 – algoritmi realiseerimine

- **Algoritm kui programm**
 - operatsioonid – andmeosa
 - järjestus ja tingimuse – juhtosa
- **GCD (Greatest Common Divisor) – suurim ühistegur**

```
while ( x != y ) {
    if ( x < y ) y = y - x;
    else      x = x - y;
}
```

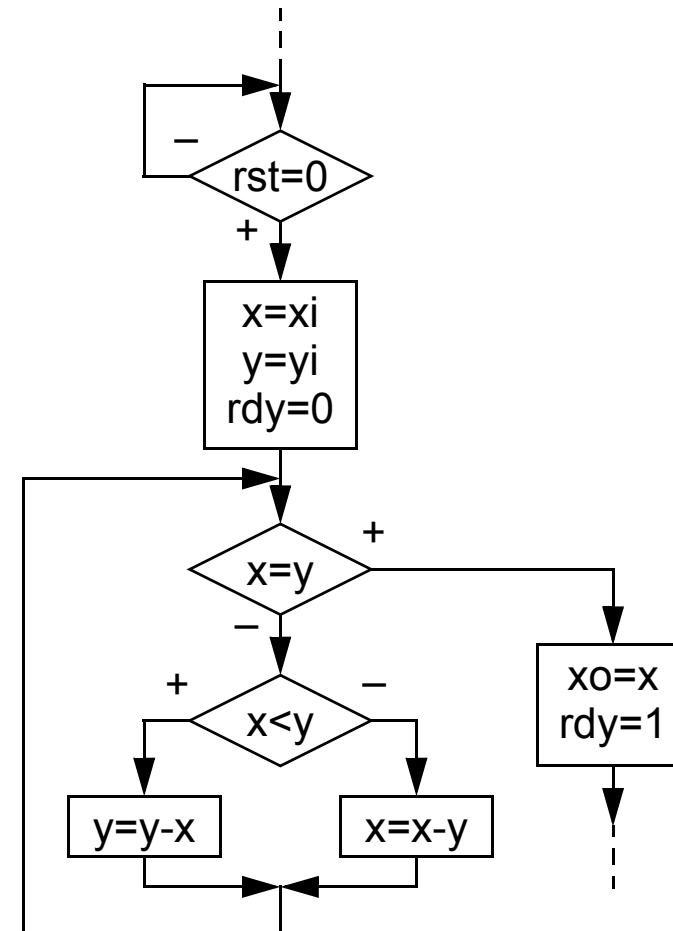
 - operatsioonid – võrdlused ja lahutamine
 - konkreetse operatsiooni realiseerimine?
 - $A < B \implies A - B < 0$ / $A \neq B \implies A - B \neq 0$ – kõike saab teha lahutajaga?!
 - sama riistvara kõikidele operatsioonidele või korraga arvutamine?
 - kiirus või suurus? [ja kas see ongi probleem?]

GCD (Greatest Common Divisor)

- **Algoritm**
 - andmete sisestamine
 - arvutamine
 - tulemuste väljastamine

```
while ( rst != 0 );
x = xi;    y = yi;    rdy = 0;
while ( x != y ) {
    if ( x < y ) y = y - x;
    else      x = x - y;
}
xo = x;    rdy = 1;
```

- **Asünkroonne sisend-väljund!**
 - taktsignaali vajalik





GCD – käitumuslik kirjeldus

- Spetsifikatsioon ~~ käitumuslik kirjeldus
 - sisendi/väljundi ajastus fikseeritud – juht- ja takt-signaali

```
process -- gcd-bhv.vhdl
  variable x, y: unsigned(15 downto 0);
begin
  -- Wait for the new input data
  wait on clk until clk='1' and rst='0';
  x := xi;    y := yi;    rdy <= '0';
  wait on clk until clk='1';
  -- Calculate
  while x /= y loop
    if x < y then y := y - x;
    else        x := x - y;    end if;
  end loop;
  -- Ready
  xo <= x;    rdy <= '1';
  wait on clk until clk='1';
end process;
```

Probleemid

- tsüklis puudub takt
- keerukas “wait” käsk
(- mitu “wait” käsku)

Mida proovida?

- erinevad süntesaatorid
- suuruse minimeerimine
- kiiruse tõstmine

Tehnoloogiad – ASIC, FPGA

VHDL kood & testpingid

<http://mini.pld.ttu.ee/~lrv/gcd/>

GCD – sünteesitav kood?

- Takteeritud käitumuslik stiil

```
process -- gcd-bhvc.vhdl
  variable x, y: unsigned(15 downto 0);
begin
  -- Wait for the new input data
  while rst = '1' loop
    wait on clk until clk='1';
  end loop;
  x := xi;    y := yi;    rdy <= '0';
  wait on clk until clk='1';
  -- Calculate
  while x /= y loop
    if x < y then y := y - x;
    else        x := x - y;    end if;
    wait on clk until clk='1';
  end loop;
  -- Ready
  xo <= x;    rdy <= '1';
  wait on clk until clk='1';
end process;
```

ASIC: sünteesitav

961 e.g. / 20.0 ns

2 lahutajat, 2 võrdlejat

FPGA: mitte-sünteesitav

“wait” käsud tsüklis :(
juhtautomaat vajalik :(

Võimalikud valikud

- funktsioonide jagamine
- universaalsed funktsioonid
- spekulatiivne arvutamine